OOP2

Generated by Doxygen 1.11.0

Chapter 1

README

V1.5 Pridėta bazinė klasė "zmogus":

```
class zmogus {
protected:
    string var = " ", pav = " ";
public:
    virtual void setvar(const string& vardas) = 0;
    virtual string getvar() const = 0;
    virtual void setpav(const string& pavarde) = 0;
    virtual string getpav() const = 0;

    zmogus() = default;
    virtual ~zmogus() {};

    zmogus(const zmogus& laikStud);  // Copy constructor
    zmogus(zmogus&& laikStud) noexcept;  // Move constructor
    zmogus& operator=(const zmogus& laikStud);  // Copy assignment operator
    zmogus& operator=(zmogus&& laikStud) noexcept;  // Move assignment operator
};
```

kaedangi klasė yra abstrakti, naujas objektas tipo "zmogus", nėra leistinas:

Šiai versijai taip pat pritaikyta penkių metodų taisyklė, funkcijos su zmogaus tipo kintamaisiais perkeltos į klasę "zmogus".

V1.2 RULE OF FIVE

Rule of five:

1. Destructor (Destruktorius) - išvalo atmintį, kad nebūtų duomenų nutekėjimo

```
mok::~mok() {
    nd.clear();
}
```

1. Copy Constructor (Kopijabvimo konstruktorius) - leidžia saugiai kopijuoti visą objektą

```
mok::mok(const mok& laikStud) {
    var = laikStud.var;
    pav = laikStud.pav;
    eg = laikStud.eg;
    gal_vid = laikStud.gal_vid;
    gal_med = laikStud.gal_med;
    nd = laikStud.nd;
}
```

2. Copy Assignment Operator (Kopijavimo priskyrimo operatorius) - leidžia jau sukurtam objektui priskirti reikšmes iš kito objekto

```
mok& mok::operator=(const mok& laikStud) {
    if (this != &laikStud) {
        var = laikStud.var;
        pav = laikStud.pav;
        eg = laikStud.eg;
        gal_vid = laikStud.gal_vid;
        gal_med = laikStud.gal_med;
        nd = laikStud.nd;
    }
    return *this;
}
```

3. Move Constructor (Perkėlimo konstruktorius) - leidžia perkelti duomenis iš vieno objekto į kitą
```
mok::mok(mok&& laikStud) noexcept
:   var(move(laikStud.var)),
    pav(move(laikStud.pav)),
    eg(laikStud.eg),
    gal_vid(laikStud.gal_vid),
    gal_med(laikStud.gal_med),
    nd(move(laikStud.nd))
{
    laikStud.eg = 0;
    laikStud.gal_vid = 0.0;
    laikStud.gal_med = 0.0;
}
```

4. Move Assignment Operator (Perkįlimo priskyrimo operatorius) - panašus į perkėlimo konstruktorių, bet naudojamas kai duomenys turi būti perkelti į jau egzistuojantį objektą
```
mok& mok::operator=(mok&& laikStud) noexcept {
    if (this != &laikStud) {
        var = move(laikStud.var);
        pav = move(laikStud.pav);
        eg = laikStud.eg;
        gal_vid = laikStud.gal_vid;
        gal_med = laikStud.gal_med;
        nd = move(laikStud.nd);
        laikStud.eg = 0;
        laikStud.gal_vid = 0.0;
        laikStud.gal_med = 0.0;
    }
    return *this;
}
```

Įvesties ir išvesties persidengimo metodai:

Jei rašome pvz.: cout << studentas; - bus išvedami visi objekto "studentas" kintamieji
```
ostream& operator«(ostream& output, const mok& stud) {
    output « stud.getvar() « " " « stud.getpav() « " " « stud.geteg() « " ";
    vector<int> pazymiai = stud.getnd();
    for (int pazymys : pazymiai) {
        output « pazymys « " ";
    }
    return output;
}
```

Jei rašome pvz.: cin >> studentas; - vartotojas turės švesti visus kintamuosius, kurie turi būti objekte "studentas"
```
istream& operator»(istream& input, mok& stud) {
    string vardas, pavarde;
    int pazymys, egzaminas;
    vector<int> namuD;
    input » vardas » pavarde » egzaminas;
    stud.setvar(vardas);
    stud.setpav(pavarde);
    stud.seteg(egzaminas);
    stud.getnd().clear();
    while (input » pazymys) {
        namuD.push_back(pazymys);
    }
    stud.setnd(namuD);
    return input;
}
```

V1.1 TESTAVIMAS

CLASS VS STRUCT

struct:

class:

Kaip matome, programa su struktūra veikia greičiau.

OPTIMIZAVIMAS

KAIP VEIKIA PROGRAMA

Ši programa skaičiuoja studentų galutinį tam tikro kurso balą, naudodama studento pažymius ir egzaminų rezultatus. Programa realizuota su trimis skirtingais konteineriais: vector, deque, list; ir su trejomis skirtingomis strategijomis, todėl galima pasirinkti, kaip norima dirbti. Galima duomenis įvedinėti ranka, galima ir liepti programai juos skaityti iš failo. Štai taip veikia paleista programa:

1. Paleidus programą išvedamas klausimas, kaip norima, kad būtų nuskaityti duomenys - ar iš failo, ar įvesti ranka;
2. Jei pasirenkame duomenis įrašyti ranka, išvedamas pasirinkimo meniu: ar įvedinėjame ranka, ar leidžiame programai sugeneruoti pažymius automatiškai, ar norime kad tiek studentų vardai, tiek jų pažymiai būtų generuojami auromatiškai;
3. Jei pasirinkame, kad duomenys būtų skaitomi iš failo svarbu, kad failas būtų įkeltas į tą patį aplanką kartu su programa;
4. Vedant vis naujus duomenis, programa paklaus jai rūpimų klausimų, kurie būtini galutinio balo skaičiavimui;
5. Jei buvo pasirinkta duomenis skaityti iš failo, reiks pasirinkti, ir kur juos išvesti - ar į ekraną, ar į naujai sugeneruotą failą.

V1.0 TESTAVIMAS

Testavimo sistemos parametrai:

PIRMA TESTAVIMO DALIS

Rezultatu apibendrinimas ir palyginimas:

ANTRA TESTAVIMO DALIS

1 STRATEGIJA:

2 STRATEGIJA

3 STRATEGIJA

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

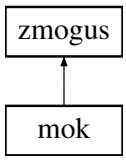
Here is a list of all files with brief descriptions:

Chapter 5

Class Documentation

5.1 mok Class Reference

Isvestine klase, aprasanti studenta.
`#include <studentas.h>`
Inheritance diagram for mok:



Public Member Functions

- void setvar (const string &vardas)

  < *Galutinis balas, apskaiciuotas naudojant mediana*
- string getvar () const

  Grazina varda.
- void setpav (const string &pavarde)

  Nustato pavarde.
- string getpav () const

  Grazina pavarde.
- void seteg (int egzaminas)

  Nustato egzamino bala.
- int geteg () const

  Grazina egzamino bala.
- void setgal_vid (double Gal_vid)

  Nustato galutini bala, apskaiciuota naudojant vidurki.
- double getgal_vid () const

  Grazina galutini bala, apskaiciuota naudojant vidurki.
- void setgal_med (double Gal_med)

  Nustato galutini bala, apskaiciuota naudojant mediana.
- double getgal_med () const

  Grazina galutini bala, apskaiciuota naudojant mediana.
- void setnd (const vector< int > &ND)

  Nustato namu darbu pazymius.
- vector< int > getnd () const

  Grazina namu darbu pazymius.
- void isvalymas ()

Isvalo objekto duomenis.

- mok ()=default

  Numatytasis konstruktorius.
- ~mok ()

  Destruktorius.
- mok (const mok &laikStud)

  Kopijavimo konstruktorius.
- mok & operator= (const mok &laikStud)

  Kopijavimo priskyrimo operatorius.
- mok (mok &&laikStud) noexcept

  Perkelimo konstruktorius.
- mok & operator= (mok &&laikStud) noexcept

  Perkelimo priskyrimo operatorius.

Public Member Functions inherited from zmogus

- zmogus ()=default

  Numatytasis konstruktorius.
- virtual ~zmogus ()

  Destruktorius.
- zmogus (const zmogus &laikStud)

  Kopijavimo konstruktorius.
- zmogus (zmogus &&laikStud) noexcept

  Perkelimo konstruktorius.
- zmogus & operator= (const zmogus &laikStud)

  Kopijavimo priskyrimo operatorius.
- zmogus & operator= (zmogus &&laikStud) noexcept

  Perkelimo priskyrimo operatorius.

Friends

- ostream & operator<< (ostream &output, const mok &stud)

  Operatorius srauto isvedimui.
- istream & operator>> (istream &input, mok &stud)

  Operatorius srauto ivedimui.

Additional Inherited Members

Protected Attributes inherited from zmogus

- string var = " "

  Zmogaus vardas.
- string pav = " "

  Zmogaus pavarde.

5.1.1 Detailed Description

Isvestine klase, aprasanti studenta.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 mok() [1/3]

```
mok::mok ()  [default]
```

Numatytasis konstruktorius.

5.1.2.2 ~mok()

```
mok::~mok ()
```
Destruktorius.

5.1.2.3 mok() [2/3]

```
mok::mok (
            const mok & laikStud)
```
Kopijavimo konstruktorius.

Parameters

laikStud	Objektas, is kurio kopijuojama.

5.1.2.4 mok() [3/3]

```
mok::mok (
            mok && laikStud)  [noexcept]
```
Perkelimo konstruktorius.

Parameters

laikStud	Objektas, is kurio perkeliama.

5.1.3 Member Function Documentation

5.1.3.1 geteg()

```
int mok::geteg () const
```
Grazina egzamino bala.

Returns

Egzamino balas.

5.1.3.2 getgal_med()

```
double mok::getgal_med () const
```
Grazina galutini bala, apskaiciuota naudojant mediana.

Returns

Galutinis balas, apskaiciuotas maudojant mediana.

5.1.3.3 getgal_vid()

```
double mok::getgal_vid () const
```
Grazina galutini bala, apskaiciuota naudojant vidurki.

Returns

Galutinis balas, apskaiciuotas maudojant vidurki.

5.1.3.4 getnd()

```
vector< int > mok::getnd () const
```

Grazina namu darbu pazymius.

Returns

Namu darbu pazymiai.

5.1.3.5 getpav()

```
string mok::getpav () const  [virtual]
```

Grazina pavarde.

Returns

Pavarde.

Implements zmogus.

5.1.3.6 getvar()

```
string mok::getvar () const  [virtual]
```

Grazina varda.

Returns

Vardas.

Implements zmogus.

5.1.3.7 isvalymas()

```
void mok::isvalymas ()
```

Isvalo objekto duomenis.

5.1.3.8 operator=() [1/2]

```
mok & mok::operator= (
            const mok & laikStud)
```

Kopijavimo priskyrimo operatorius.

Parameters

laikStud	Objektas, is kurio kopijuojama.

Returns

Priskirtas objektas.

5.1.3.9 operator=() [2/2]

```
mok & mok::operator= (
            mok && laikStud)  [noexcept]
```

Perkelimo priskyrimo operatorius.

Parameters

laikStud	Objektas, is kurio perkeliama.

Returns

Priskirtas objektas.

5.1.3.10 seteg()

```
void mok::seteg (
            int egzaminas)
```

Nustato egzamino bala.

Parameters

egzaminas	Egzamino balas.

5.1.3.11 setgal_med()

```
void mok::setgal_med (
            double Gal_med)
```

Nustato galutini bala, apskaiciuota naudojant mediana.

Parameters

Gal_vid	Galutinis balas, apskaiciuotas naudojant mediana.

5.1.3.12 setgal_vid()

```
void mok::setgal_vid (
            double Gal_vid)
```

Nustato galutini bala, apskaiciuota naudojant vidurki.

Parameters

Gal_vid	Galutinis balas, apskaiciuotas naudojant vidurki.

5.1.3.13 setnd()

```
void mok::setnd (
            const vector< int > & ND)
```

Nustato namu darbu pazymius.

Parameters

ND	Namu darbu pazymiai.

5.1.3.14 setpav()

```
void mok::setpav (
            const string & pavarde)  [virtual]
```

Nustato pavarde.

Parameters

pavarde	Pavarde.

Implements zmogus.

5.1.3.15 setvar()

```
void mok::setvar (
            const string & vardas)  [virtual]
```

< Galutinis balas, apskaiciuotas naudojant mediana
Nustato varda.

Parameters

vardas	Vardas.

Implements zmogus.

5.1.4 Friends And Related Symbol Documentation

5.1.4.1 operator<<

```
ostream & operator<< (
            ostream & output,
            const mok & stud)  [friend]
```

Operatorius srauto isvedimui.

Parameters

output	Srautas.
stud	Studentas.

Returns

Srauto objektas.

5.1.4.2 operator>>

```
istream & operator>> (
            istream & input,
            mok & stud)  [friend]
```

Operatorius srauto ivedimui.

Parameters

input	Srautas.
stud	Studentas.

Returns

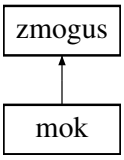
Srauto objektas.

The documentation for this class was generated from the following files:

- studentas.h
- studentas.cpp

5.2 zmogus Class Reference

Abstrakti klase, sukurta aprasyti zmogu.
`#include <studentas.h>`
Inheritance diagram for zmogus:



Public Member Functions

- virtual void setvar (const string &vardas)=0
  Nustato varda.
- virtual string getvar () const =0
  Grazina varda.
- virtual void setpav (const string &pavarde)=0
  Nustato pavarde.
- virtual string getpav () const =0
  Grazina pavarde.
- zmogus ()=default
  Numatytasis konstruktorius.
- virtual ~zmogus ()
  Destruktorius.
- zmogus (const zmogus &laikStud)
  Kopijavimo konstruktorius.
- zmogus (zmogus &&laikStud) noexcept
  Perkelimo konstruktorius.
- zmogus & operator= (const zmogus &laikStud)
  Kopijavimo priskyrimo operatorius.
- zmogus & operator= (zmogus &&laikStud) noexcept
  Perkelimo priskyrimo operatorius.

Protected Attributes

- string var = " "
  Zmogaus vardas.
- string pav = " "
  Zmogaus pavarde.

5.2.1 Detailed Description

Abstrakti klase, sukurta aprasyti zmogu.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 zmogus() [1/3]

`zmogus::zmogus () [default]`
Numatytasis konstruktorius.

5.2.2.2 ~zmogus()

`virtual zmogus::~zmogus () [inline], [virtual]`
Destruktorius.

5.2.2.3 zmogus() [2/3]

```
zmogus::zmogus (
            const zmogus & laikStud)
```

Kopijavimo konstruktorius.

Parameters

laikStud	Objektas, is kurio kopijuojama.

5.2.2.4 zmogus() [3/3]

```
zmogus::zmogus (
            zmogus && laikStud)  [noexcept]
```

Perkelimo konstruktorius.

Parameters

laikStud	Objektas, is kurio perkeliama.

5.2.3 Member Function Documentation

5.2.3.1 getpav()

```
virtual string zmogus::getpav () const  [pure virtual]
```

Grazina pavarde.

Returns

Pavarde.

Implemented in mok.

5.2.3.2 getvar()

```
virtual string zmogus::getvar () const  [pure virtual]
```

Grazina varda.

Returns

Vardas.

Implemented in mok.

5.2.3.3 operator=() [1/2]

```
zmogus & zmogus::operator= (
            const zmogus & laikStud)
```

Kopijavimo priskyrimo operatorius.

Parameters

laikStud	Objektas, is kurio kopijuojama.

Returns

Priskirtas objektas.

5.2.3.4 operator=() [2/2]

```
zmogus & zmogus::operator= (
            zmogus && laikStud)  [noexcept]
```

Perkelimo priskyrimo operatorius.

Parameters

laikStud	Objektas, is kurio perkeliama.

Returns

Priskirtas objektas.

5.2.3.5 setpav()

```
virtual void zmogus::setpav (
            const string & pavarde)  [pure virtual]
```

Nustato pavarde.

Parameters

pavarde	Pavarde.

Implemented in mok.

5.2.3.6 setvar()

```
virtual void zmogus::setvar (
            const string & vardas)  [pure virtual]
```

Nustato varda.

Parameters

vardas	Vardas.

Implemented in mok.

5.2.4 Member Data Documentation

5.2.4.1 pav

```
string zmogus::pav = " "  [protected]
```

Zmogaus pavarde.

5.2.4.2 var

```
string zmogus::var = " "  [protected]
```

Zmogaus vardas.

The documentation for this class was generated from the following files:

- studentas.h
- studentas.cpp

Chapter 6

File Documentation

6.1 funkcijos.cpp File Reference

```
#include "studentas.h"
#include "funkcijos.h"
```

Functions

- void ivedimas (vector< mok > &stud)

  Ivedimo funkcija.
- void meniu (int &antrasPasirinkimas)

  Pirmas meniu.
- void calculateResults (vector< mok > &stud)

  Rezultatu skaiciavimo funkcija.
- char rikiavimoklausimas ()

  Rikiavimo klausimas.
- void isvedimas (vector< mok > &stud, ostream &os, char a)

  Isvedimo funkcija.
- void meniuAntras (int &antrasPasirinkimas)

  Antras meniu.
- bool pagalVarda (const mok &a, const mok &b)

  Rikiavimo funkcija pagal varda.
- bool pagalPavarde (const mok &a, const mok &b)

  Rikiavimo funkcija pagal pavarde.
- bool pagalMediana (const mok &a, const mok &b)

  Rikiavimo funkcija pagal mediana.
- bool pagalVidurki (const mok &a, const mok &b)

  Rikiavimo funkcija pagal vidurki.
- milliseconds trukmesSkaiciavimas (high_resolution_clock::time_point pradzia, high_resolution_clock::time↩ _point pabaiga)

  Trukmes skaiciavimo funkcija.
- void failuGeneravimas (int studentuKiekis, const string &failoPavadinimas)

  Failu generavimo funkcija.
- void konteineriai (int studentuKiekis, vector< mok > &studentai, char a, vector< mok > &vargsiukai)

  Skirstymo i konteinerius funkcija.
- void isvalymas (vector< mok > &vektorius)

  Vektoriaus isvalymo funkcija.
- void failuNuskaitymas (vector< mok > &studentai, string &failoPavadinimas)

  Failu nuskaitymo funkcija.

- int pirmasP (int &pirmasPasirinkimas)

  Klausimas naudotojui.
- int treciasP (int &treciasPasirinkimas)

  Klausimas naudotojui.
- void rikiavimas (int ketvirtasPasirinkimas, vector< mok > &studentai)

  Rikiavimo funkcija.
- void testavimoRezultatai (bool result, const string &testas)

  Testavimo rezultatu funkcija.

6.1.1 Function Documentation

6.1.1.1 calculateResults()

```
void calculateResults (
            vector< mok > & stud)
```

Rezultatu skaiciavimo funkcija.

Parameters

stud	Studentu vektorius.

6.1.1.2 failuGeneravimas()

```
void failuGeneravimas (
            int studentuKiekis,
            const string & failoPavadinimas)
```

Failu generavimo funkcija.

Parameters

studentuKiekis	Studentu Kiekis.
failoPavadinimas	Failo pavadinimas.

6.1.1.3 failuNuskaitymas()

```
void failuNuskaitymas (
            vector< mok > & studentai,
            string & failoPavadinimas)
```

Failu nuskaitymo funkcija.

Parameters

studentai	Studentu vektorius.
failoPavadinimas	Failo pavadinimas.

6.1.1.4 isvalymas()

```
void isvalymas (
            vector< mok > & vektorius)
```

Vektoriaus isvalymo funkcija.

Parameters

vektorius	Vektorius, kuri reikia isvalyti.

6.1.1.5 isvedimas()

```
void isvedimas (
            vector< mok > & stud,
            ostream & os,
            char a)
```

Isvedimo funkcija.
Si funkcija naudojama ir isvedimui i faila, ir i ekrana.

Parameters

stud	Studentu vektorius.
os	Ivesties srautas.
a	Rikiavimo tipa nurodantis simbolis.

6.1.1.6 ivedimas()

```
void ivedimas (
            vector< mok > & stud)
```

Ivedimo funkcija.
Si funkcija naudojama tik pasirinkus ivedima ranka.

Parameters

stud	Studentu vektorius.

6.1.1.7 konteineriai()

```
void konteineriai (
            int studentuKiekis,
            vector< mok > & studentai,
            char a,
            vector< mok > & vargsiukai)
```

Skirstymo i konteinerius funkcija.

Parameters

studentuKiekis	Studentu kiekis.
studentai	Studentu vektorius.
a	Rikiavimo tipa nurodantis simbolis.
vargsiukai	Vargsiuku vektorius.

6.1.1.8 meniu()

```
void meniu (
            int & antrasPasirinkimas)
```

Pirmas meniu.
Sis meniu atsiranda tada, kada naudotojas pasirenka duomenis ivedineti ranka.

Parameters

antrasPasirinkimas	Naudotojo pasirinkimas.

6.1.1.9 meniuAntras()

```
void meniuAntras (
            int & ketvirtasPasirinkimas)
```

Antras meniu.
Naudotojas renkasi, pagal ka isrikiuotus duomenis nori matyti.

Parameters

ketvirtasPasirinkimas	Naudotojo pasirinkimas.

6.1.1.10 pagalMediana()

```
bool pagalMediana (
            const mok & a,
            const mok & b)
```

Rikiavimo funkcija pagal mediana.

Parameters

a	Pirmas mok objektas.
b	Antras mok objektas.

Returns

Tiesa, jei pirma mediana yra mazesne uz antra.

6.1.1.11 pagalPavarde()

```
bool pagalPavarde (
            const mok & a,
            const mok & b)
```

Rikiavimo funkcija pagal pavarde.

Parameters

a	Pirmas mok objektas.
b	Antras mok objektas.

Returns

Tiesa, jei pirma pavarde yra zemiau uz antra.

6.1.1.12 pagalVarda()

```
bool pagalVarda (
            const mok & a,
            const mok & b)
```

Rikiavimo funkcija pagal varda.

Parameters

a	Pirmas mok objektas.
b	Antras mok objektas.

Returns

Tiesa, jei pirmas vardas yra zemiau uz antra.

6.1.1.13 pagalVidurki()

```
bool pagalVidurki (
            const mok & a,
            const mok & b)
```

Rikiavimo funkcija pagal vidurki.

Parameters

a	Pirmas mok objektas.
b	Antras mok objektas.

Returns

Tiesa, jei pirmas vidurkis yra mazesnis uz antra.

6.1.1.14 pirmasP()

```
int pirmasP (
            int & pirmasPasirinkimas)
```

Klausimas naudotojui.
Funkcija, kuri klausia naudotojo ar jis nori duomenis ivesti ranka, ar nuskaityti is failo.

Parameters

pirmasPasirinkimas	Naudotojo pasirinkimas.

Returns

Naudotojo pasirinkimas.

6.1.1.15 rikiavimas()

```
void rikiavimas (
            const int ketvirtaspasirinkimas,
            vector< mok > & studentai)
```

Rikiavimo funkcija.

Parameters

ketvirtaspasirinkimas	Ketvirtas pasirinkimas.
studentai	Studentu vektorius.

6.1.1.16 rikiavimoklausimas()

```
char rikiavimoklausimas ()
```

Rikiavimo klausimas.

Returns

Simbolis, nurodantis pagal ka bus rikiuojami duomenys.

6.1.1.17 testavimoRezultatai()

```
void testavimoRezultatai (
            bool result,
            const string & testas)
```

Testavimo rezultatu funkcija.

Parameters

result	Testo rezultatas.
testas	Testo pavadinimas.

6.1.1.18 treciasP()

```
int treciasP (
            int & treciasPasirinkimas)
```

Klausimas naudotojui.
Funkcija, kuri klausia naudotojo ar jis nori, kad duomenys butu isvesti ekrane, ar i faila.

Parameters

treciasPasirinkimas	Naudotojo pasirinkimas.

Returns

Naudotojo pasirinkimas.

6.1.1.19 trukmesSkaiciavimas()

```
milliseconds trukmesSkaiciavimas (
            high_resolution_clock::time_point pradzia,
            high_resolution_clock::time_point pabaiga)
```

Trukmes skaiciavimo funkcija.

Parameters

pradzia	Pradzios laikas.
pabaiga	Pabaigos laikas.

Returns

Trukmes intervalas milisekundemis.

6.2 funkcijos.h File Reference

siame faile aprasytos kode naudojamos funkcijos

```
#include <iostream>
#include <iomanip>
#include <vector>
#include <algorithm>
#include <numeric>
#include <ctime>
#include <fstream>
#include <sstream>
#include <string>
#include <chrono>
#include <cstdlib>
#include <cassert>
```

Functions

- void ivedimas (vector< mok > &stud)

  Ivedimo funkcija.
- void calculateResults (vector< mok > &stud)

  Rezultatu skaiciavimo funkcija.
- char rikiavimoklausimas ()

  Rikiavimo klausimas.
- void isvedimas (vector< mok > &stud, ostream &os, char a)

  Isvedimo funkcija.
- void meniu (int &antrasPasirinkimas)

  Pirmas meniu.
- void meniuAntras (int &ketvirtasPasirinkimas)

  Antras meniu.
- bool pagalVarda (const mok &a, const mok &b)

  Rikiavimo funkcija pagal varda.
- bool pagalPavarde (const mok &a, const mok &b)

  Rikiavimo funkcija pagal pavarde.
- bool pagalMediana (const mok &a, const mok &b)

  Rikiavimo funkcija pagal mediana.
- bool pagalVidurki (const mok &a, const mok &b)

  Rikiavimo funkcija pagal vidurki.
- milliseconds trukmesSkaiciavimas (high_resolution_clock::time_point pradzia, high_resolution_clock::time_point pabaiga)

  Trukmes skaiciavimo funkcija.
- void failuGeneravimas (int studentuKiekis, const string &failoPavadinimas)

  Failu generavimo funkcija.
- void konteineriai (int studentuKiekis, vector< mok > &studentai, char a, vector< mok > &vargsiukai)

  Skirstymo i konteinerius funkcija.
- void isvalymas (vector< mok > &vektorius)

  Vektoriaus isvalymo funkcija.
- void failuNuskaitymas (vector< mok > &studentai, string &failoPavadinimas)

  Failu nuskaitymo funkcija.
- int pirmasP (int &pirmasPasirinkimas)

  Klausimas naudotojui.
- int treciasP (int &treciasPasirinkimas)

  Klausimas naudotojui.
- void rikiavimas (const int ketvirtaspasirinkimas, vector< mok > &studentai)

  Rikiavimo funkcija.
- void testavimoRezultatai (bool result, const string &testas)

  Testavimo rezultatu funkcija.

6.2.1 Detailed Description

siame faile aprasytos kode naudojamos funkcijos

6.2.2 Function Documentation

6.2.2.1 calculateResults()

```
void calculateResults (
            vector< mok > & stud)
```

Rezultatu skaiciavimo funkcija.

Parameters

stud	Studentu vektorius.

6.2.2.2 failuGeneravimas()

```
void failuGeneravimas (
            int studentuKiekis,
            const string & failoPavadinimas)
```

Failu generavimo funkcija.

Parameters

studentuKiekis	Studentu Kiekis.
failoPavadinimas	Failo pavadinimas.

6.2.2.3 failuNuskaitymas()

```
void failuNuskaitymas (
            vector< mok > & studentai,
            string & failoPavadinimas)
```

Failu nuskaitymo funkcija.

Parameters

studentai	Studentu vektorius.
failoPavadinimas	Failo pavadinimas.

6.2.2.4 isvalymas()

```
void isvalymas (
            vector< mok > & vektorius)
```

Vektoriaus isvalymo funkcija.

Parameters

vektorius	Vektorius, kuri reikia isvalyti.

6.2.2.5 isvedimas()

```
void isvedimas (
            vector< mok > & stud,
            ostream & os,
            char a)
```

Isvedimo funkcija.
Si funkcija naudojama ir isvedimui i faila, ir i ekrana.

Parameters

stud	Studentu vektorius.
os	Ivesties srautas.
a	Rikiavimo tipa nurodantis simbolis.

6.2.2.6 ivedimas()

```
void ivedimas (
            vector< mok > & stud)
```

Ivedimo funkcija.
Si funkcija naudojama tik pasirinkus ivedima ranka.

Parameters

stud	Studentu vektorius.

6.2.2.7 konteineriai()

```
void konteineriai (
            int studentuKiekis,
            vector< mok > & studentai,
            char a,
            vector< mok > & vargsiukai)
```

Skirstymo i konteinerius funkcija.

Parameters

studentuKiekis	Studentu kiekis.
studentai	Studentu vektorius.
a	Rikiavimo tipa nurodantis simbolis.
vargsiukai	Vargsiuku vektorius.

6.2.2.8 meniu()

```
void meniu (
            int & antrasPasirinkimas)
```

Pirmas meniu.
Sis meniu atsiranda tada, kada naudotojas pasirenka duomenis ivedineti ranka.

Parameters

antrasPasirinkimas	Naudotojo pasirinkimas.

6.2.2.9 meniuAntras()

```
void meniuAntras (
            int & ketvirtasPasirinkimas)
```

Antras meniu.
Naudotojas renkasi, pagal ka isrikiuotus duomenis nori matyti.

Parameters

ketvirtasPasirinkimas	Naudotojo pasirinkimas.

6.2.2.10 pagalMediana()

```
bool pagalMediana (
            const mok & a,
            const mok & b)
```

Rikiavimo funkcija pagal mediana.

Parameters

a	Pirmas mok objektas.
b	Antras mok objektas.

Returns

Tiesa, jei pirma mediana yra mazesne uz antra.

6.2.2.11 pagalPavarde()

```
bool pagalPavarde (
            const mok & a,
            const mok & b)
```

Rikiavimo funkcija pagal pavarde.

Parameters

a	Pirmas mok objektas.
b	Antras mok objektas.

Returns

Tiesa, jei pirma pavarde yra zemiau uz antra.

6.2.2.12 pagalVarda()

```
bool pagalVarda (
            const mok & a,
            const mok & b)
```

Rikiavimo funkcija pagal varda.

Parameters

a	Pirmas mok objektas.
b	Antras mok objektas.

Returns

Tiesa, jei pirmas vardas yra zemiau uz antra.

6.2.2.13 pagalVidurki()

```
bool pagalVidurki (
            const mok & a,
            const mok & b)
```

Rikiavimo funkcija pagal vidurki.

Parameters

a	Pirmas mok objektas.
b	Antras mok objektas.

Returns

Tiesa, jei pirmas vidurkis yra mazesnis uz antra.

6.2.2.14 pirmasP()

```
int pirmasP (
            int & pirmasPasirinkimas)
```

Klausimas naudotojui.
Funkcija, kuri klausia naudotojo ar jis nori duomenis ivesti ranka, ar nuskaityti is failo.

Parameters

pirmasPasirinkimas	Naudotojo pasirinkimas.

Returns

Naudotojo pasirinkimas.

6.2.2.15 rikiavimas()

```
void rikiavimas (
            const int ketvirtaspasirinkimas,
            vector< mok > & studentai)
```

Rikiavimo funkcija.

Parameters

ketvirtaspasirinkimas	Ketvirtas pasirinkimas.
studentai	Studentu vektorius.

6.2.2.16 rikiavimoklausimas()

```
char rikiavimoklausimas ()
```

Rikiavimo klausimas.

Returns

Simbolis, nurodantis pagal ka bus rikiuojami duomenys.

6.2.2.17 testavimoRezultatai()

```
void testavimoRezultatai (
            bool result,
            const string & testas)
```

Testavimo rezultatu funkcija.

Parameters

result	Testo rezultatas.
testas	Testo pavadinimas.

6.2.2.18 treciasP()

```
int treciasP (
            int & treciasPasirinkimas)
```

Klausimas naudotojui.
Funkcija, kuri klausia naudotojo ar jis nori, kad duomenys butu isvesti ekrane, ar i faila.

Parameters

treciasPasirinkimas	Naudotojo pasirinkimas.

Returns

Naudotojo pasirinkimas.

6.2.2.19 trukmesSkaiciavimas()

```
milliseconds trukmesSkaiciavimas (
            high_resolution_clock::time_point pradzia,
            high_resolution_clock::time_point pabaiga)
```

Trukmes skaiciavimo funkcija.

Parameters

pradzia	Pradzios laikas.
pabaiga	Pabaigos laikas.

Returns

Trukmes intervalas milisekundemis.

6.3 funkcijos.h

Go to the documentation of this file.

```
00001
00006 #ifndef FUNKCIJOS_H
00007 #define FUNKCIJOS_H
00008
00009 #include <iostream>
00010 #include <iomanip>
00011 #include <vector>
00012 #include <algorithm>
00013 #include <numeric>
00014 #include <ctime>
00015 #include <fstream>
00016 #include <sstream>
00017 #include <string>
00018 #include <chrono>
00019 #include <cstdlib>
00020 #include <cassert>
00021
00022 using namespace std;
00023 using namespace std::chrono;
00024
00030 void ivedimas(vector<mok>& stud);
00031
00036 void calculateResults(vector<mok>& stud);
00037
00042 char rikiavimoklausimas();
00043
00051 void isvedimas(vector<mok>& stud, ostream& os, char a);
00052
00058 void meniu(int& antrasPasirinkimas);
00059
00065 void meniuAntras(int& ketvirtasPasirinkimas);
00066
00073 bool pagalVarda(const mok& a, const mok& b);
00074
00081 bool pagalPavarde(const mok& a, const mok& b);
00082
00089 bool pagalMediana(const mok& a, const mok& b);
00090
00097 bool pagalVidurki(const mok& a, const mok& b);
00098
00105 milliseconds trukmesSkaiciavimas(high_resolution_clock::time_point pradzia,
      high_resolution_clock::time_point pabaiga);
00106
00112 void failuGeneravimas(int studentuKiekis, const string& failoPavadinimas);
```

```
00113
00121 void konteineriai(int studentuKiekis, vector<mok>& studentai, char a, vector<mok>& vargsiukai);
00122
00127 void isvalymas(vector<mok>& vektorius);
00128
00134 void failuNuskaitymas(vector<mok>& studentai, string& failoPavadinimas);
00135
00142 int pirmasP(int& pirmasPasirinkimas);
00143
00150 int treciasP(int& treciasPasirinkimas);
00151
00157 void rikiavimas(const int ketvirtaspasirinkimas, vector<mok>& studentai);
00158
00164 void testavimoRezultatai(bool result, const string& testas);
00165
00166 #endif
```

6.4 OOP.cpp File Reference

```
#include "studentas.h"
#include "funkcijos.h"
```

Functions

- int main ()

6.4.1 Function Documentation

6.4.1.1 main()

```
int main ()
```

6.5 out/build/x64-debug/CMakeFiles/3.28.0-msvc1/CompilerIdC/CMake↩ CCompilerId.c File Reference

Macros

- #define __has_include(x) 0
- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define C_VERSION

Functions

- int main (int argc, char ∗argv[ ])

Variables

- char const ∗ info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const ∗ info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const ∗ info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char ∗ info_language_standard_default
- const char ∗ info_language_extensions_default

6.5.1 Macro Definition Documentation

6.5.1.1 __has_include

```
#define __has_include(
              x) 0
```

6.5.1.2 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

6.5.1.3 C_VERSION

```
#define C_VERSION
```

6.5.1.4 COMPILER_ID

```
#define COMPILER_ID ""
```

6.5.1.5 DEC

```
#define DEC(
              n)
```

Value:

```
  ('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

6.5.1.6 HEX

```
#define HEX(
              n)
```

Value:

```
  ('0' + ((n)»28 & 0xF)), \
  ('0' + ((n)»24 & 0xF)), \
  ('0' + ((n)»20 & 0xF)), \
  ('0' + ((n)»16 & 0xF)), \
  ('0' + ((n)»12 & 0xF)), \
  ('0' + ((n)»8  & 0xF)), \
  ('0' + ((n)»4  & 0xF)), \
  ('0' + ((n)     & 0xF))
```

6.5.1.7 PLATFORM_ID

```
#define PLATFORM_ID
```

6.5.1.8 STRINGIFY

```
#define STRINGIFY(
              X) STRINGIFY_HELPER(X)
```

6.5.1.9 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(
              X) #X
```

6.5.2 Function Documentation

6.5.2.1 main()

```
int main (
            int argc,
            char * argv[])
```

6.5.3 Variable Documentation

6.5.3.1 info_arch

```
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
```

6.5.3.2 info_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

6.5.3.3 info_language_extensions_default

```
const char* info_language_extensions_default
```

Initial value:

```
= "INFO" ":" "extensions_default["



  "OFF"

"]"
```

6.5.3.4 info_language_standard_default

```
const char* info_language_standard_default
```

Initial value:

```
=
  "INFO" ":" "standard_default[" C_VERSION "]"
```

6.5.3.5 info_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

6.6 out/build/x64-debug/CMakeFiles/3.28.0-msvc1/CompilerIdCXX/↩ CMakeCXXCompilerId.cpp File Reference

Macros

- #define __has_include(x) 0
- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define CXX_STD __cplusplus

Functions

- int main (int argc, char ∗argv[])

Variables

- char const * info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const * info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const * info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char * info_language_standard_default
- const char * info_language_extensions_default

6.6.1 Macro Definition Documentation

6.6.1.1 __has_include

```
#define __has_include(
              x) 0
```

6.6.1.2 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

6.6.1.3 COMPILER_ID

```
#define COMPILER_ID ""
```

6.6.1.4 CXX_STD

```
#define CXX_STD __cplusplus
```

6.6.1.5 DEC

```
#define DEC(
              n)
```

Value:

```
  ('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

6.6.1.6 HEX

```
#define HEX(
              n)
```

Value:

```
  ('0' + ((n)>>28 & 0xF)), \
  ('0' + ((n)>>24 & 0xF)), \
  ('0' + ((n)>>20 & 0xF)), \
  ('0' + ((n)>>16 & 0xF)), \
  ('0' + ((n)>>12 & 0xF)), \
  ('0' + ((n)>>8  & 0xF)), \
  ('0' + ((n)>>4  & 0xF)), \
  ('0' + ((n)     & 0xF))
```

6.6.1.7 PLATFORM_ID

```
#define PLATFORM_ID
```

6.6.1.8 STRINGIFY

```
#define STRINGIFY(
              X) STRINGIFY_HELPER(X)
```

6.6.1.9 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(
              X) #X
```

6.6.2 Function Documentation

6.6.2.1 main()

```
int main (
             int argc,
             char * argv[])
```

6.6.3 Variable Documentation

6.6.3.1 info_arch

```
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
```

6.6.3.2 info_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

6.6.3.3 info_language_extensions_default

```
const char* info_language_extensions_default
```

Initial value:

```
= "INFO" ":" "extensions_default["



  "OFF"

"]"
```

6.6.3.4 info_language_standard_default

```
const char* info_language_standard_default
```

Initial value:

```
= "INFO" ":" "standard_default["









  "98"

"]"
```

6.6.3.5 info_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

6.7 out/build/x64-debug/CMakeFiles/ShowIncludes/foo.h File Reference

6.8 foo.h

Go to the documentation of this file.

```
00001
```

6.9 out/build/x64-debug/CMakeFiles/ShowIncludes/main.c File Reference

```
#include "foo.h"
```

Functions

- int main ()

6.9.1 Function Documentation

6.9.1.1 main()

```
int main ()
```

6.10 README.md File Reference

6.11 studentas.cpp File Reference

```
#include "studentas.h"
#include "funkcijos.h"
```

Functions

- ostream & operator<< (ostream &output, const mok &stud)
- istream & operator>> (istream &input, mok &stud)

6.11.1 Function Documentation

6.11.1.1 operator<<()

```
ostream & operator<< (
            ostream & output,
            const mok & stud)
```

Parameters

output	Srautas.
stud	Studentas.

Returns

Srauto objektas.

6.11.1.2 operator>>()

```
istream & operator>> (
            istream & input,
            mok & stud)
```

Parameters

input	Srautas.
stud	Studentas.

Returns

Srauto objektas.

6.12 studentas.h File Reference

siame faile aprasytos studento ir zmogaus klases

```
#include <iostream>
#include <vector>
#include <string>
```

Classes

- class zmogus

  Abstrakti klase, sukurta aprasyti zmogu.
- class mok

  Isvestine klase, aprasanti studenta.

6.12.1 Detailed Description

siame faile aprasytos studento ir zmogaus klases

6.13 studentas.h

Go to the documentation of this file.

```
00001
00007 #ifndef STUDENTAS_H
00008 #define STUDENTAS_H
00009
00010 #include <iostream>
00011 #include <vector>
00012 #include <string>
00013
00014 using namespace std;
00015
00016
00022 class zmogus {
00023 protected:
00024     string var = " ";
00025     string pav = " ";
00026 public:
00031     virtual void setvar(const string& vardas) = 0;
00032
00037     virtual string getvar() const = 0;
00038
00043     virtual void setpav(const string& pavarde) = 0;
00044
00049     virtual string getpav() const = 0;
00050
00054     zmogus() = default;
00055
00059     virtual ~zmogus() {};
00060
00065     zmogus(const zmogus& laikStud);  // Copy constructor
00066
00071     zmogus(zmogus&& laikStud) noexcept;  // Move constructor
00072
00078     zmogus& operator=(const zmogus& laikStud);  // Copy assignment operator
00079
00085     zmogus& operator=(zmogus&& laikStud) noexcept;  // Move assignment operator
00086 };
00087
00092 class mok : public zmogus {
00093 private:
00094     int eg = 0;
00095     vector<int> nd = { 0 };
00096     double gal_vid = 0.0;
00097     double gal_med = 0.0;
00098 public:
00103     void setvar(const string& vardas);
00104
00109     string getvar() const;
```

```
00110
00115     void setpav(const string& pavarde);
00116
00121     string getpav() const;
00122
00127     void seteg(int egzaminas);
00128
00133     int geteg()const;
00134
00139     void setgal_vid(double Gal_vid);
00140
00145     double getgal_vid() const;
00146
00151     void setgal_med(double Gal_med);
00152
00157     double getgal_med() const;
00158
00163     void setnd(const vector<int>& ND);
00164
00169     vector<int> getnd() const;
00170
00174     void isvalymas();
00175     // RULE OF FIVE:
00176
00180     mok() = default;
00181
00185     ~mok();
00186
00191     mok(const mok& laikStud);
00192
00198     mok& operator=(const mok& laikStud);
00199
00204     mok(mok&& laikStud) noexcept;
00205
00211     mok& operator=(mok&& laikStud) noexcept;
00212
00219     friend ostream& operator« (ostream& output, const mok& stud);
00220
00227     friend istream& operator»(istream& input, mok& stud);
00228
00229 };
00230
00231 #endif
```