Studentu pazymiu skaiciavimo programa

Generated by Doxygen 1.13.2

Chapter 1

README

Programa leidžia įvesti arba sugeneruoti studentų duomenis bei pažymius, apskaičiuoti jų galutinius pažymius su vidurkiu arba mediana pasirinktinai, taip pat suskirstyti juos į du konteinerius, "protingi" ir "neprotingi", pagal vidurkį.

1.1 Programos paleidimo instrukcija:

1. Kompiuteryje susiinstaliuoti `make`
2. Parsisiųsti relizą ir unzippinti parsisiųstus failus
3. Per komandinę eilutę nunaviguoti į relizo direktoriją ir įvesti "make run"

Paleidus programą naudotojui parodomas meniu su 7 pasirinkimais:

1 - Suvesti duomenis ranka : Leidžia suvesti tiek studentų vardus, tiek pažymius ranka

2 - Sugeneruoti pažymius: Leidžia įvesti studentų vardus, o pažymius programa sugeneruoja atsitiktinai

3 - Sugeneruoti pažymius, vardus ir pavardes: Programa sugeneruoja tiek studentų vardus, tiek pažymius

4 - Nuskaityti duomenis iš failo: Studentų duomenys yra nuskaitomi iš pasirinkto failo

5 - Testuoti kodą ir išvesti 3 laikų vidurkį: Nuskaito pasirinktą studentų failą ir parodo, kiek laiko užtruko tai padaryti

6 - Tirti funkcijas: suteikia galimybę atlikti du tyrimus: pirmas tyrimas sugeneruoja kelis skirtingų dydžių studentų failus ir parodo generavimo trukmę, o antras tyrimas perskaito sugeneruotus studentų failus ir ištiria jų nuskaitymo bei išskirstymo į protingus ir neprotingus studentus pagal pažymį spartą

7 - Tirti konstruktorius: Ištestuoja visų studentų klasės konstruktorių ir perkrautų operatorių veikimą

8 - Baigti darbą: Išjungia programą

1.2 v1.5

Programoje pridėta Žmogaus klasė, kurioje saugomas tik žmogaus vardas ir pavardė, o Studentų klasė sukurta jos pagrindu, taigi turi Žmogaus vardą ir pavardę, tačiau papildomai dar turi namų darbų ir egzamino pažymius. Visi ankstesni Studentų klasės metodai toliau su ja veikia taip pat, kaip ir anksčiau.

Taip pat Žmogaus klasė yra abstrakti, taigi jos tipo objektų sukurti negalima.

1.3 v1.2

Studentų klasei pridėti rule of five konstruktoriai, kurie leidžia sukurti naujus objektus kopijuojant ar perkeliant senus. Taip pat yra perkrauti įvesties ir išvesties operatoriai - įvesties operatorius pritaikomas nuskaitant studentų duomenis iš failo, o išvesties operatorius pritaikomas išvedant studento vardą ir balą tiek į terminalą, tiek į failą.

Tiek konstruktoriai, tiek perkrauti operatoriai yra testuojami naujoje meniu funkcijoje, kuri gražina teiginius apie sėkmingą testavimą.

1.4 Programos testavimas naudojant 6 funkciją:

Prieš pirmą kartą atliekant antrą tyrimą, privaloma bent kartą atlikti pirmą, kad būtų sugeneruoti failai, su kuriais būtų galima atlikti antrą tyrimą.

Testavimui naudojamo kompiuterio specifikacijos:

Procesorius	Intel(R) Core(TM) Ultra 7 155H
HDD	SSD
RAM	SK Hynix 4GB x 8

1.5 Klasės

Optimizavimo vėliavėlė	Greitis su 100 000	Greitis su 1 000 000	.exe failo dydis
Jokios	1.12708 s.	8.48943 s.	809 KB
-O1	0.708478 s.	5.06565 s.	537 KB
-O2	0.693832 s.	4.76938 s.	483 KB
-O3	0.669088 s.	4.57423 s.	479 KB

1.6 Struktūros

Optimizavimo vėliavėlė	Greitis su 100 000	Greitis su 1 000 000	.exe failo dydis
Jokios	0.705166 s.	5.59301 s.	636 KB
-O1	0.530952 s.	3.37178 s.	457 KB
-O2	0.451381 s.	3.50529 s.	437 KB
-O3	0.498676 s.	3.5854 s.	442 KB

1.7 Tyrimas su vektoriais:

	Pirma strategija	Antra strategija	Trečia strategija
1000	0.0346398 s.	0.0321874 s.	0.0271154 s.
10000	0.0588841 s.	0.0484614 s.	0.0532457 s.
100000	0.75594 s.	0.729521 s.	0.72437 s.
1000000	5.16762 s.	4.71562 s.	4.56433 s.
10000000	44.4975 s.	41.9814 s.	40.5489 s.

1.8 Tyrimas su sąrašais:

	Pirma strategija	Antra strategija	Trečia strategija
1000	0.0360853 s.	0.0349864 s.	0.0335232 s.
10000	0.0414789 s.	0.0462905 s.	0.0493538 s.
100000	0.769735 s.	0.675944 s.	0.717202 s.
1000000	5.61882 s.	5.16409 s.	6.04569 s.
10000000	48.6259 s.	47.7247 s.	53.2475 s.

1.9 Tyrimas su dėklais:

	Pirma strategija	Antra strategija	Trečia strategija
1000	0.0351316 s.	0.0222539 s.	0.0299696 s.
10000	0.072327 s.	0.0456836 s.	0.0608322 s.
100000	0.835998 s.	0.772752 s.	0.718334 s.
1000000	6.35227 s.	5.51153 s.	5.34346 s.
10000000	52.4223 s.	46.8919 s.	47.6157 s.

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:

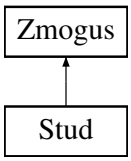
Chapter 5

Class Documentation

5.1 Stud Class Reference

`#include <klase.h>`

Inheritance diagram for Stud:



Public Member Functions

- Stud ()
- Stud (const string &var, const string &pav, const int &egz, const vector< int > &nd_)
- Stud (const string &var, const string &pav, const int &egz)
- Stud (const Stud &s)
- Stud & operator= (const Stud &s)
- Stud (Stud &&s)
- Stud & operator= (Stud &&s)
- ~Stud ()
- void setEgzaminas (const double &egz)
- void setND (const vector< int > &nd_)
- void setPazymys (const int &paz)
- void setGalutinisSuVidurkiu (const double &vid)
- void setGalutinisSuMediana (const double &med)
- const double getEgzaminas () const
- vector< int > getND () const
- int getPazymys (int &i) const
- double getGalutinisSuVidurkiu () const
- double getGalutinisSuMediana () const
- bool operator== (const Stud &s) const
- bool operator!= (const Stud &s) const

Public Member Functions inherited from Zmogus

- Zmogus ()
- Zmogus (const string &var, const string &pav)
- virtual ~Zmogus ()=0
- void setVardas (const string &var)
- void setPavarde (const string &pav)
- const string getVardas () const
- const string getPavarde () const
- Zmogus (Zmogus &&z)
- Zmogus & operator= (Zmogus &&z)

Friends

- istream & operator>> (istream &is, Stud &s)
- ostream & operator<< (ostream &os, Stud &s)

Additional Inherited Members

Protected Attributes inherited from Zmogus

- string vardas
- string pavarde

5.1.1 Constructor & Destructor Documentation

5.1.1.1 Stud() [1/5]

```
Stud::Stud ()  [inline]
```

5.1.1.2 Stud() [2/5]

```
Stud::Stud (
            const string & var,
            const string & pav,
            const int & egz,
            const vector< int > & nd_)  [inline]
```

5.1.1.3 Stud() [3/5]

```
Stud::Stud (
            const string & var,
            const string & pav,
            const int & egz)  [inline]
```

5.1.1.4 Stud() [4/5]

```
Stud::Stud (
            const Stud & s)
```

5.1.1.5 Stud() [5/5]

```
Stud::Stud (
            Stud && s)
```

5.1.1.6 ~Stud()

```
Stud::~Stud ()  [inline]
```

5.1.2 Member Function Documentation

5.1.2.1 getEgzaminas()

```
const double Stud::getEgzaminas () const  [inline]
```

5.1.2.2 getGalutinisSuMediana()

```
double Stud::getGalutinisSuMediana () const  [inline]
```

5.1.2.3 getGalutinisSuVidurkiu()

```
double Stud::getGalutinisSuVidurkiu () const  [inline]
```

5.1.2.4 getND()

```
vector< int > Stud::getND () const  [inline]
```

5.1.2.5 getPazymys()

```
int Stud::getPazymys (
            int & i) const  [inline]
```

5.1.2.6 operator"!=()

```
bool Stud::operator!= (
            const Stud & s) const  [inline]
```

5.1.2.7 operator=() [1/2]

```
Stud & Stud::operator= (
            const Stud & s)
```

5.1.2.8 operator=() [2/2]

```
Stud & Stud::operator= (
            Stud && s)
```

5.1.2.9 operator==()

```
bool Stud::operator== (
            const Stud & s) const  [inline]
```

5.1.2.10 setEgzaminas()

```
void Stud::setEgzaminas (
            const double & egz)  [inline]
```

5.1.2.11 setGalutinisSuMediana()

```
void Stud::setGalutinisSuMediana (
            const double & med)  [inline]
```

5.1.2.12 setGalutinisSuVidurkiu()

```
void Stud::setGalutinisSuVidurkiu (
            const double & vid)  [inline]
```

5.1.2.13 setND()

```
void Stud::setND (
            const vector< int > & nd_)  [inline]
```

5.1.2.14 setPazymys()

```
void Stud::setPazymys (
            const int & paz)  [inline]
```

5.1.3 Friends And Related Symbol Documentation

5.1.3.1 operator<<

```
ostream & operator<< (
            ostream & os,
            Stud & s)  [friend]
```

5.1.3.2 operator>>

```
istream & operator>> (
            istream & is,
            Stud & s)  [friend]
```

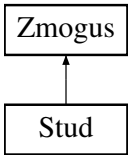
The documentation for this class was generated from the following files:

- headers/klase.h
- klase.cpp

5.2 Zmogus Class Reference

```
#include <klase.h>
```

Inheritance diagram for Zmogus:



Public Member Functions

- Zmogus ()
- Zmogus (const string &var, const string &pav)
- virtual ~Zmogus ()=0
- void setVardas (const string &var)
- void setPavarde (const string &pav)
- const string getVardas () const
- const string getPavarde () const
- Zmogus (Zmogus &&z)
- Zmogus & operator= (Zmogus &&z)

Protected Attributes

- string vardas
- string pavarde

5.2.1 Constructor & Destructor Documentation

5.2.1.1 Zmogus() [1/3]

```
Zmogus::Zmogus ()  [inline]
```

5.2.1.2 Zmogus() [2/3]

```
Zmogus::Zmogus (
            const string & var,
            const string & pav)  [inline]
```

5.2.1.3 ~Zmogus()

```
Zmogus::~Zmogus ()  [pure virtual]
```

5.2.1.4 Zmogus() [3/3]

```
Zmogus::Zmogus (
            Zmogus && z)  [inline]
```

5.2.2 Member Function Documentation

5.2.2.1 getPavarde()

```
const string Zmogus::getPavarde () const  [inline]
```

5.2.2.2 getVardas()

```
const string Zmogus::getVardas () const  [inline]
```

5.2.2.3 operator=()

```
Zmogus & Zmogus::operator= (
            Zmogus && z)  [inline]
```

5.2.2.4 setPavarde()

```
void Zmogus::setPavarde (
            const string & pav)  [inline]
```

5.2.2.5 setVardas()

```
void Zmogus::setVardas (
            const string & var)  [inline]
```

5.2.3 Member Data Documentation

5.2.3.1 pavarde

```
string Zmogus::pavarde  [protected]
```

5.2.3.2 vardas

```
string Zmogus::vardas  [protected]
```

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

- headers/klase.h
- klase.cpp

Chapter 6

File Documentation

6.1 funkcijos.cpp File Reference

```
#include "headers/antrastesSuTemplates.h"
```

Functions

- void vardoIvedimas (Stud &laik)
- void pazymiuIvedimas (Stud &laik)
- void vardoGeneravimas (Stud &laik)
- void pazymiuGeneravimas (Stud &laik)
- bool pagalVarda (Stud &a, Stud &b)
- bool pagalPavarde (Stud &a, Stud &b)
- bool pagalVidurki (Stud &a, Stud &b)
- bool pagalMediana (Stud &a, Stud &b)
- void testas (string failoPavadinimas)
- int rusiavimoPasirinkimas ()
- string failoPasirinkimas (string klausimas)
- void failoGeneravimas (int failoIlgis)
- int galutinioBaloPasirinkimas ()
- double vidurkis (vector< int > nd)
- double mediana (vector< int > nd)
- void constructorTestas ()
- void copyConstructorTestas ()
- void copyAssignmentTestas ()
- void moveConstructorTestas ()
- void moveAssignmentTestas ()
- void ivestiesTestas ()
- void isvestiesTestas ()
- void destruktoriausTestas ()

6.1.1 Function Documentation

6.1.1.1 constructorTestas()

```
void constructorTestas ()
```

6.1.1.2 copyAssignmentTestas()

```
void copyAssignmentTestas ()
```

6.1.1.3 copyConstructorTestas()

```
void copyConstructorTestas ()
```

6.1.1.4 destruktoriausTestas()

```
void destruktoriausTestas ()
```

6.1.1.5 failoGeneravimas()

```
void failoGeneravimas (
            int failoIlgis)
```

6.1.1.6 failoPasirinkimas()

```
string failoPasirinkimas (
            string klausimas)
```

6.1.1.7 galutinioBaloPasirinkimas()

```
int galutinioBaloPasirinkimas ()
```

6.1.1.8 isvestiesTestas()

```
void isvestiesTestas ()
```

6.1.1.9 ivestiesTestas()

```
void ivestiesTestas ()
```

6.1.1.10 mediana()

```
double mediana (
            vector< int > nd)
```

6.1.1.11 moveAssignmentTestas()

```
void moveAssignmentTestas ()
```

6.1.1.12 moveConstructorTestas()

```
void moveConstructorTestas ()
```

6.1.1.13 pagalMediana()

```
bool pagalMediana (
            Stud & a,
            Stud & b)
```

6.1.1.14 pagalPavarde()

```
bool pagalPavarde (
            Stud & a,
            Stud & b)
```

6.1.1.15 pagalVarda()

```
bool pagalVarda (
            Stud & a,
            Stud & b)
```

6.1.1.16 pagalVidurki()

```
bool pagalVidurki (
            Stud & a,
            Stud & b)
```

6.1.1.17 pazymiuGeneravimas()

```
void pazymiuGeneravimas (
            Stud & laik)
```

6.1.1.18 pazymiuIvedimas()

```
void pazymiuIvedimas (
            Stud & laik)
```

6.1.1.19 rusiavimoPasirinkimas()

```
int rusiavimoPasirinkimas ()
```

6.1.1.20 testas()

```
void testas (
            string failoPavadinimas)
```

6.1.1.21 vardoGeneravimas()

```
void vardoGeneravimas (
            Stud & laik)
```

6.1.1.22 vardoIvedimas()

```
void vardoIvedimas (
            Stud & laik)
```

6.1.1.23 vidurkis()

```
double vidurkis (
            vector< int > nd)
```

6.2 headers/antrastesBeTemplates.h File Reference

```
#include "klase.h"
```

Functions

- void vardoIvedimas (Stud &laik)
- void pazymiuIvedimas (Stud &laik)
- void vardoGeneravimas (Stud &laik)
- void pazymiuGeneravimas (Stud &laik)
- bool pagalVarda (Stud &a, Stud &b)
- bool pagalPavarde (Stud &a, Stud &b)
- bool pagalVidurki (Stud &a, Stud &b)
- bool pagalMediana (Stud &a, Stud &b)
- void testas (string failoPavadinimas)
- int rusiavimoPasirinkimas ()
- string failoPasirinkimas (string klausimas)
- void failoGeneravimas (int failoIlgis)
- int galutinioBaloPasirinkimas ()
- void constructorTestas ()
- void copyConstructorTestas ()
- void copyAssignmentTestas ()
- void moveConstructorTestas ()
- void moveAssignmentTestas ()
- void ivestiesTestas ()
- void isvestiesTestas ()
- void destruktoriausTestas ()

6.2.1 Function Documentation

6.2.1.1 constructorTestas()

```
void constructorTestas ()
```

6.2.1.2 copyAssignmentTestas()

```
void copyAssignmentTestas ()
```

6.2.1.3 copyConstructorTestas()

```
void copyConstructorTestas ()
```

6.2.1.4 destruktoriausTestas()

```
void destruktoriausTestas ()
```

6.2.1.5 failoGeneravimas()

```
void failoGeneravimas (
            int failoIlgis)
```

6.2.1.6 failoPasirinkimas()

```
string failoPasirinkimas (
            string klausimas)
```

6.2.1.7 galutinioBaloPasirinkimas()

```
int galutinioBaloPasirinkimas ()
```

6.2.1.8 isvestiesTestas()

```
void isvestiesTestas ()
```

6.2.1.9 ivestiesTestas()

```
void ivestiesTestas ()
```

6.2.1.10 moveAssignmentTestas()

```
void moveAssignmentTestas ()
```

6.2.1.11 moveConstructorTestas()

```
void moveConstructorTestas ()
```

6.2.1.12 pagalMediana()

```
bool pagalMediana (
            Stud & a,
            Stud & b)
```

6.2.1.13 pagalPavarde()

```
bool pagalPavarde (
            Stud & a,
            Stud & b)
```

6.2.1.14 pagalVarda()

```
bool pagalVarda (
            Stud & a,
            Stud & b)
```

6.2.1.15 pagalVidurki()

```
bool pagalVidurki (
            Stud & a,
            Stud & b)
```

6.2.1.16 pazymiuGeneravimas()

```
void pazymiuGeneravimas (
            Stud & laik)
```

6.2.1.17 pazymiuIvedimas()

```
void pazymiuIvedimas (
            Stud & laik)
```

6.2.1.18 rusiavimoPasirinkimas()

```
int rusiavimoPasirinkimas ()
```

6.2.1.19 testas()

```
void testas (
            string failoPavadinimas)
```

6.2.1.20 vardoGeneravimas()

```
void vardoGeneravimas (
            Stud & laik)
```

6.2.1.21 vardoIvedimas()

```
void vardoIvedimas (
            Stud & laik)
```

6.3 antrastesBeTemplates.h

Go to the documentation of this file.

```
00001 #include "klase.h"
00002 void vardoIvedimas(Stud &laik);
00003 void pazymiuIvedimas(Stud &laik);
00004 void vardoGeneravimas(Stud &laik);
00005 void pazymiuGeneravimas(Stud &laik);
00006 bool pagalVarda(Stud &a, Stud &b);
00007 bool pagalPavarde(Stud &a, Stud &b);
00008 bool pagalVidurki(Stud &a, Stud &b);
00009 bool pagalMediana(Stud &a, Stud &b);
00010 void testas(string failoPavadinimas);
00011 int rusiavimoPasirinkimas();
00012 string failoPasirinkimas(string klausimas);
00013 void failoGeneravimas(int failoIlgis);
00014 int galutinioBaloPasirinkimas();
00015 void constructorTestas();
00016 void copyConstructorTestas();
00017 void copyAssignmentTestas();
00018 void moveConstructorTestas();
00019 void moveAssignmentTestas();
00020 void ivestiesTestas();
00021 void isvestiesTestas();
00022 void destruktoriausTestas();
```

6.4 headers/antrastesSuTemplates.h File Reference

```
#include "mano_lib.h"
#include "antrastesBeTemplates.h"
```

Functions

- template<typename Container>
  void ivedimas (Container &studentai, int &meniuPasirinkimas)
- template<typename Container>
  void nuskaitymasSuBuferiu (Container &studentai, string failoPavadinimas)
- template<typename Container>
  void isvedimas (Container &studentai, int galutinioBaloPasirinkimas, ostream &isvedimoBudas)
- template<typename Container>
  void rusiavimas (Container &studentai, int pasirinkimas)
- template<typename Container>
  void tyrimas (Container &studentai)
- template<typename Container>
  void pirmaStrategija (Container &studentai, Container &protingi, Container &neprotingi, int galutinisBalas)
- template<typename Container>
  void antraStrategija (Container &studentai, Container &neprotingi)
- template<typename Container>
  void treciaStrategija (Container &studentai, Container &neprotingi)

6.4.1 Function Documentation

6.4.1.1 antraStrategija()

```
template<typename Container>
void antraStrategija (
            Container & studentai,
            Container & neprotingi)
```

6.4.1.2 isvedimas()

```
template<typename Container>
void isvedimas (
            Container & studentai,
            int galutinioBaloPasirinkimas,
            ostream & isvedimoBudas)
```

6.4.1.3 ivedimas()

```
template<typename Container>
void ivedimas (
            Container & studentai,
            int & meniuPasirinkimas)
```

6.4.1.4 nuskaitymasSuBuferiu()

```
template<typename Container>
void nuskaitymasSuBuferiu (
            Container & studentai,
            string failoPavadinimas)
```

6.4.1.5 pirmaStrategija()

```
template<typename Container>
void pirmaStrategija (
             Container & studentai,
             Container & protingi,
             Container & neprotingi,
             int galutinisBalas)
```

6.4.1.6 rusiavimas()

```
template<typename Container>
void rusiavimas (
             Container & studentai,
             int pasirinkimas)
```

6.4.1.7 treciaStrategija()

```
template<typename Container>
void treciaStrategija (
             Container & studentai,
             Container & neprotingi)
```

6.4.1.8 tyrimas()

```
template<typename Container>
void tyrimas (
             Container & studentai)
```

6.5 antrastesSuTemplates.h

Go to the documentation of this file.

```
00001 #include "mano_lib.h"
00002 #include "antrastesBeTemplates.h"
00003
00004 template <typename Container>
00005 void ivedimas(Container &studentai, int &meniuPasirinkimas)
00006 {
00007     bool naujasStudentas = true;
00008     int pasirinkimas;
00009     while (naujasStudentas)
00010     {
00011         Stud laik;
00012         if (meniuPasirinkimas == 1 || meniuPasirinkimas == 2)
00013             vardoIvedimas(laik);
00014         if (meniuPasirinkimas == 1)
00015             pazymiuIvedimas(laik);
00016         else if (meniuPasirinkimas == 2)
00017             pazymiuGeneravimas(laik);
00018         else if (meniuPasirinkimas == 3)
00019         {
00020             vardoGeneravimas(laik);
00021             pazymiuGeneravimas(laik);
00022         }
00023         laik.setGalutinisSuVidurkiu((vidurkis(laik.getND()) * 0.4) + (laik.getEgzaminas() * 0.6));
00024         laik.setGalutinisSuMediana((mediana(laik.getND()) * 0.4) + (laik.getEgzaminas() * 0.6));
00025         studentai.push_back(laik);
00026         while (true)
```

```
00027         {
00028             try
00029             {
00030                 cout « "Ar norite įvesti dar vieną studentą?" « endl;
00031                 cout « "1 - taip" « endl;
00032                 cout « "2 - ne" « endl;
00033                 cin » pasirinkimas;
00034                 if (cin.fail())
00035                 {
00036                     cin.clear();
00037                     cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
00038                     throw "Įvedėte ne skaičių!";
00039                 }
00040                 else if (pasirinkimas < 1 || pasirinkimas > 2)
00041                     throw "Įvedėte netinkamą skaičių!";
00042                 else
00043                     break;
00044             }
00045             catch (const char *e)
00046             {
00047                 cout « e « endl;
00048                 continue;
00049             }
00050         }
00051         if (pasirinkimas == 2)
00052             naujasStudentas = false;
00053     }
00054 }
00055
00056 template <typename Container>
00057 void nuskaitymasSuBuferiu(Container &studentai, string failoPavadinimas)
00058 {
00059     vector<string> laik;
00060     string eilute;
00061     stringstream buferis;
00062     ifstream fin;
00063     while (true)
00064     {
00065         try
00066         {
00067             fin.open(failoPavadinimas);
00068             if (fin.fail())
00069             {
00070                 throw "Nepavyko atidaryti failo.";
00071             }
00072             else
00073                 break;
00074         }
00075         catch (const char *e)
00076         {
00077             cout « e « endl;
00078             failoPavadinimas = failoPasirinkimas("Pasirinkite kitą failą: ");
00079             continue;
00080         }
00081     }
00082     buferis « fin.rdbuf();
00083     fin.close();
00084
00085     while (buferis)
00086     {
00087         if (!buferis.eof())
00088         {
00089             getline(buferis, eilute);
00090             laik.push_back(eilute);
00091         }
00092         else
00093             break;
00094     }
00095     studentai.clear();
00096     int skaicius;
00097     bool pirmaEilute = true;
00098     for (string a : laik)
00099     {
00100         string vardas, pavarde;
00101         istringstream is(a);
00102         Stud laikStudentas;
00103         if (pirmaEilute == true)
00104             pirmaEilute = false;
00105         else
00106         {
00107             is » laikStudentas;
00108             studentai.push_back(move(laikStudentas));
00109         }
00110     }
00111 }
00112
00113 template <typename Container>
```

```
00114 void isvedimas(Container &studentai, int galutinioBaloPasirinkimas, ostream &isvedimoBudas)
00115 {
00116     stringstream buferis;
00117     buferis « setw(16) « left « "Vardas" « setw(16) « "Pavarde";
00118     if (galutinioBaloPasirinkimas == 1)
00119         buferis « setw(20) « "Galutinis (Vid.)" « endl;
00120     else
00121         buferis « setw(20) « "Galutinis (Med.)" « endl;
00122     buferis « "-------------------------------------------------------------" « endl;
00123     for (Stud i : studentai)
00124     {
00125         buferis « i;
00126     }
00127     isvedimoBudas « buferis.rdbuf();
00128     studentai.clear();
00129 }
00130
00131 template <typename Container>
00132 void rusiavimas(Container &studentai, int pasirinkimas)
00133 {
00134     if constexpr (std::is_same_v<Container, list<Stud»)
00135     {
00136         if (pasirinkimas == 1)
00137             studentai.sort(pagalVarda);
00138         else if (pasirinkimas == 2)
00139             studentai.sort(pagalPavarde);
00140         else if (pasirinkimas == 3)
00141             studentai.sort(pagalVidurki);
00142         else if (pasirinkimas == 4)
00143             studentai.sort(pagalMediana);
00144     }
00145     else
00146     {
00147         if (pasirinkimas == 1)
00148             sort(studentai.begin(), studentai.end(), pagalVarda);
00149         if (pasirinkimas == 2)
00150             sort(studentai.begin(), studentai.end(), pagalPavarde);
00151         if (pasirinkimas == 3)
00152             sort(studentai.begin(), studentai.end(), pagalVidurki);
00153         if (pasirinkimas == 4)
00154             sort(studentai.begin(), studentai.end(), pagalMediana);
00155     }
00156 }
00157
00158 template <typename Container>
00159 void tyrimas(Container &studentai)
00160 {
00161     int tyrimoPasirinkimas, rusPasirinkimas, kiekioPasirinkimas, galBaloPasirinkimas,
      skirstymoPasirinkimas;
00162     string sugeneruotasFailas;
00163     while (true)
00164     {
00165         try
00166         {
00167             cout « "Kuri tyrima nori atlikti?" « endl;
00168             cout « "1" « endl;
00169             cout « "2" « endl;
00170             cin » tyrimoPasirinkimas;
00171             if (cin.fail())
00172             {
00173                 cin.clear();
00174                 cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
00175                 throw "Įvedėte ne skaičių!";
00176             }
00177             else if (tyrimoPasirinkimas < 1 || tyrimoPasirinkimas > 2)
00178                 throw "Įvedėte netinkamą skaičių!";
00179             else
00180                 break;
00181         }
00182         catch (const char *e)
00183         {
00184             cout « e « endl;
00185             continue;
00186         }
00187     }
00188     while (true)
00189     {
00190         try
00191         {
00192             cout « "Kuria rusiavimo strategija noretum naudoti?" « endl;
00193             cout « "1" « endl;
00194             cout « "2" « endl;
00195             cout « "3" « endl;
00196             cin » skirstymoPasirinkimas;
00197             if (cin.fail())
00198             {
00199                 cin.clear();
```

```
00200                 cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
00201                 throw "Įvedėte ne skaičių!";
00202             }
00203             else if (skirstymoPasirinkimas < 1 || skirstymoPasirinkimas > 3)
00204                 throw "Įvedėte netinkamą skaičių!";
00205             else
00206                 break;
00207         }
00208         catch (const char *e)
00209         {
00210             cout << e << endl;
00211             continue;
00212         }
00213     }
00214     if (tyrimoPasirinkimas == 2)
00215     {
00216         rusPasirinkimas = rusiavimoPasirinkimas();
00217     }
00218     while (true)
00219     {
00220         try
00221         {
00222             cout << "Kiek kartu nori atlikti tyrima?" << endl;
00223             cin >> kiekioPasirinkimas;
00224             if (cin.fail())
00225             {
00226                 cin.clear();
00227                 cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
00228                 throw "Įvedėte ne skaičių!";
00229             }
00230             else if (kiekioPasirinkimas < 1)
00231                 throw "Įvedėte netinkamą skaičių!";
00232             else
00233                 break;
00234         }
00235         catch (const char *e)
00236         {
00237             cout << e << endl;
00238             continue;
00239         }
00240     }
00241     for (int dydzioPasirinkimas = 1000; dydzioPasirinkimas <= 10000000; dydzioPasirinkimas *= 10)
00242     {
00243         double vid = 0;
00244         for (int j = 0; j < kiekioPasirinkimas; j++)
00245         {
00246             auto pradzia = std::chrono::high_resolution_clock::now();
00247             if (tyrimoPasirinkimas == 1)
00248             {
00249                 auto t1 = std::chrono::high_resolution_clock::now();
00250                 failoGeneravimas(dydzioPasirinkimas);
00251                 auto t2 = std::chrono::high_resolution_clock::now();
00252                 cout << dydzioPasirinkimas << " studentu failo sugeneravimas truko " << (t2 - t1) / 1.0s
      << " " << endl;
00253             }
00254             if (tyrimoPasirinkimas == 2)
00255             {
00256                 Container protingi;
00257                 Container neprotingi;
00258                 studentai.clear();
00259                 auto t3 = std::chrono::high_resolution_clock::now();
00260                 nuskaitymasSuBuferiu(studentai, "studentai" + to_string(dydzioPasirinkimas) + ".txt");
00261                 auto t4 = std::chrono::high_resolution_clock::now();
00262                 cout << "Failo nuskaitymas truko: " << (t4 - t3) / 1.0s << " s." << endl;
00263                 rusiavimas(studentai, 3);
00264                 if (skirstymoPasirinkimas == 1)
00265                     pirmaStrategija(studentai, protingi, neprotingi, 1);
00266                 else if (skirstymoPasirinkimas == 2)
00267                     antraStrategija(studentai, neprotingi);
00268                 else
00269                     treciaStrategija(studentai, neprotingi);
00270                 auto t7 = std::chrono::high_resolution_clock::now();
00271                 cout << "Isskaidymas pagal galutini bala truko: " << (t7 - t4) / 1.0s << " s." << endl;
00272                 if (skirstymoPasirinkimas == 1)
00273                     rusiavimas(protingi, rusPasirinkimas);
00274                 else
00275                     rusiavimas(studentai, rusPasirinkimas);
00276                 rusiavimas(neprotingi, rusPasirinkimas);
00277                 auto t5 = std::chrono::high_resolution_clock::now();
00278                 cout << "Studentu konteinerio surusiavimas truko: " << (t5 - t7) / 1.0s << " s." << endl;
00279             }
00280             auto pabaiga = std::chrono::high_resolution_clock::now();
00281             vid += ((pabaiga - pradzia) / 1.0s);
00282         }
00283         cout << "Vidutiniskai tyrimas su " << dydzioPasirinkimas << " studentu uztruko " << vid /
      kiekioPasirinkimas << " s." << endl;
00284         cout << endl;
```

```
00285     }
00286 }
00287
00288 template <typename Container>
00289 void pirmaStrategija(Container &studentai, Container &protingi, Container &neprotingi, int
      galutinisBalas)
00290 {
00291     for (Stud s : studentai)
00292     {
00293         if (galutinisBalas == 1)
00294         {
00295             if (s.getGalutinisSuVidurkiu() >= 5)
00296                 protingi.push_back(s);
00297             else
00298                 neprotingi.push_back(s);
00299         }
00300         else
00301         {
00302             if (s.getGalutinisSuMediana() >= 5)
00303                 protingi.push_back(s);
00304             else
00305                 neprotingi.push_back(s);
00306         }
00307     }
00308     if constexpr (std::is_same_v<Container, vector<Stud»)
00309     {
00310         protingi.shrink_to_fit();
00311         neprotingi.shrink_to_fit();
00312     }
00313 }
00314
00315 template <typename Container>
00316 void antraStrategija(Container &studentai, Container &neprotingi)
00317 {
00318     while (studentai.back().getGalutinisSuVidurkiu() < 5)
00319     {
00320         neprotingi.push_back(studentai.back());
00321         studentai.pop_back();
00322     }
00323     if constexpr (std::is_same_v<Container, vector<Stud»)
00324     {
00325         neprotingi.shrink_to_fit();
00326         studentai.shrink_to_fit();
00327     }
00328 }
00329
00330 template <typename Container>
00331 void treciaStrategija(Container &studentai, Container &neprotingi)
00332 {
00333     auto it = stable_partition(studentai.begin(), studentai.end(), [](Stud &a)
00334                                { return a.getGalutinisSuVidurkiu() >= 5; });
00335     neprotingi.assign(it, studentai.end());
00336     studentai.resize(std::distance(studentai.begin(), it));
00337     if constexpr (std::is_same_v<Container, vector<Stud»)
00338     {
00339         neprotingi.shrink_to_fit();
00340         studentai.shrink_to_fit();
00341     }
00342 }
```

6.6 headers/klase.h File Reference

```
#include "mano_lib.h"
```

Classes

- class Zmogus
- class Stud

Functions

- double vidurkis (vector< int > nd)
- double mediana (vector< int > nd)

6.6.1 Function Documentation

6.6.1.1 mediana()

```
double mediana (
            vector< int > nd)
```

6.6.1.2 vidurkis()

```
double vidurkis (
            vector< int > nd)
```

6.7 klase.h

Go to the documentation of this file.

```
00001 #include "mano_lib.h"
00002 double vidurkis(vector<int> nd);
00003 double mediana(vector<int> nd);
00004 class Zmogus
00005 {
00006 protected:
00007     string vardas;
00008     string pavarde;
00009 
00010 public:
00011     Zmogus() : vardas(""), pavarde("") {}                                    // konstruktorius
00012     Zmogus(const string &var, const string &pav) : vardas(var), pavarde(pav) {} // konstruktorius su
      inputu
00013     virtual ~Zmogus() = 0;
00014     // setteriai
00015     void setVardas(const string &var) { vardas = var; };
00016     void setPavarde(const string &pav) { pavarde = pav; };
00017     // getteriai
00018     const string getVardas() const { return vardas; };
00019     const string getPavarde() const { return pavarde; };
00020 
00021     Zmogus(Zmogus &&z) : vardas(move(z.vardas)), pavarde(move(z.pavarde)) {}
00022     Zmogus &operator=(Zmogus &&z)
00023     {
00024         if (this != &z)
00025         {
00026             vardas = move(z.vardas);
00027             pavarde = move(z.pavarde);
00028         }
00029         return *this;
00030     }
00031 };
00032 class Stud : public Zmogus
00033 {
00034 private:
00035     double egzaminas;
00036     vector<int> nd;
00037     double galutinisSuVidurkiu;
00038     double galutinisSuMediana;
00039 
00040 public:
00041     Stud() : Zmogus("", ""), egzaminas(0), nd{}, galutinisSuVidurkiu(0), galutinisSuMediana(0) {}
      // konstruktorius
00042     Stud(const string &var, const string &pav, const int &egz, const vector<int> &nd_) : Zmogus(var,
      pav), egzaminas(egz), nd{nd_}, galutinisSuVidurkiu((vidurkis(nd_) * 0.4) + (egz * 0.6)),
      galutinisSuMediana((mediana(nd_) * 0.4) + (egz * 0.6)) {} // konstruktorius su inputu
00043     Stud(const string &var, const string &pav, const int &egz) : Zmogus(var, pav), egzaminas(egz),
      nd{}, galutinisSuVidurkiu((egz * 0.6)), galutinisSuMediana((egz * 0.6)) {}
      // konstruktorius su inputu
00044     Stud(const Stud &s);
00045     Stud &operator=(const Stud &s);
00046     Stud(Stud &&s);
00047     Stud &operator=(Stud &&s);
00048     ~Stud()
00049     {
00050         nd.clear();
```

```
00051         egzaminas = 0;
00052         galutinisSuVidurkiu = 0;
00053         galutinisSuMediana = 0;
00054     }
00055     friend istream &operator»(istream &is, Stud &s);
00056     friend ostream &operator«(ostream &os, Stud &s);
00057     // setteriai
00058     void setEgzaminas(const double &egz) { egzaminas = egz; };
00059     void setND(const vector<int> &nd_) { nd = nd_; };
00060     void setPazymys(const int &paz) { nd.push_back(paz); };
00061     void setGalutinisSuVidurkiu(const double &vid) { galutinisSuVidurkiu = vid; };
00062     void setGalutinisSuMediana(const double &med) { galutinisSuMediana = med; };
00063     // getteriai
00064     const double getEgzaminas() const { return egzaminas; };
00065     vector<int> getND() const { return nd; };
00066     int getPazymys(int &i) const { return nd.at(i); };
00067     double getGalutinisSuVidurkiu() const { return galutinisSuVidurkiu; };
00068     double getGalutinisSuMediana() const { return galutinisSuMediana; };
00069
00070     // papildomos funkcijos
00071     bool operator==(const Stud &s) const
00072     {
00073         if (vardas == s.vardas && pavarde == s.pavarde && egzaminas == s.egzaminas && nd == s.nd)
00074             return true;
00075         else
00076             return false;
00077     }
00078     bool operator!=(const Stud &s) const
00079     {
00080         if (vardas != s.vardas || pavarde != s.pavarde || egzaminas != s.egzaminas || nd != s.nd)
00081             return true;
00082         else
00083             return false;
00084     }
00085 };
```

6.8 headers/mano_lib.h File Reference

```
#include <bits/stdc++.h>
#include <chrono>
```

6.9 mano_lib.h

Go to the documentation of this file.

```
00001 #include <bits/stdc++.h>
00002 #include <chrono>
00003
00004 using std::cin;
00005 using std::cout;
00006 using std::endl;
00007 using std::fixed;
00008 using std::floor;
00009 using std::ifstream;
00010 using std::istringstream;
00011 using std::left;
00012 using std::setprecision;
00013 using std::setw;
00014 using std::sort;
00015 using std::string;
00016 using std::stringstream;
00017 using std::vector;
00018 using std::ws;
00019 using namespace std::literals::chrono_literals;
00020 using std::ofstream;
00021 using std::ostringstream;
00022 using std::runtime_error;
00023 using std::ostream;
00024 using std::istream;
00025 using std::to_string;
00026 using std::list;
00027 using std::deque;
00028 using std::move;
```

6.10 klase.cpp File Reference

```
#include "headers/antrastesBeTemplates.h"
```

Functions

- istream & operator>> (istream &is, Stud &s)
- ostream & operator<< (ostream &os, Stud &s)

6.10.1 Function Documentation

6.10.1.1 operator<<()

```
ostream & operator<< (
            ostream & os,
            Stud & s)
```

6.10.1.2 operator>>()

```
istream & operator>> (
            istream & is,
            Stud & s)
```

6.11 main.cpp File Reference

```
#include "headers/antrastesSuTemplates.h"
```

Functions

- int main ()

6.11.1 Function Documentation

6.11.1.1 main()

```
int main ()
```

6.12 README.md File Reference

6.13 testai.cpp File Reference

```
#include "catch.hpp"
#include "../headers/klase.h"
#include "../headers/mano_lib.h"
```

Macros

- #define CATCH_CONFIG_MAIN

Functions

- TEST_CASE ("Testuojamas konstruktorius", "[Constructor]")
- TEST_CASE ("Testuojamas kopijavimo konstruktorius", "[Copy][constructor]")
- TEST_CASE ("Testuojamas kopijavimo priskyrimo operatorius", "[Copy][assignment][operator]")
- TEST_CASE ("Testuojamas move konstruktorius", "[Move][constructor]")
- TEST_CASE ("Testuojamas move priskyrimo operatorius", "[Move][assignment][operator]")
- TEST_CASE ("Testuojamas ivedimo operatorius", "[Input][operator]")
- TEST_CASE ("Testuojamas isvedimo operatorius", "[Output][operator]")
- TEST_CASE ("Testuojamas destruktorius", "[Destructor]")
- TEST_CASE ("Testuojami getteriai", "[Getter]")

6.13.1 Macro Definition Documentation

6.13.1.1 CATCH_CONFIG_MAIN

```
#define CATCH_CONFIG_MAIN
```

6.13.2 Function Documentation

6.13.2.1 TEST_CASE() [1/9]

```
TEST_CASE (
            "Testuojamas destruktorius" ,
            "" [Destructor])
```

6.13.2.2 TEST_CASE() [2/9]

```
TEST_CASE (
            "Testuojamas isvedimo operatorius" ,
            "" [Output][operator])
```

6.13.2.3 TEST_CASE() [3/9]

```
TEST_CASE (
            "Testuojamas ivedimo operatorius" ,
            "" [Input][operator])
```

6.13.2.4 TEST_CASE() [4/9]

```
TEST_CASE (
            "Testuojamas konstruktorius" ,
            "" [Constructor])
```

6.13.2.5 TEST_CASE() [5/9]

```
TEST_CASE (
            "Testuojamas kopijavimo konstruktorius" ,
            "" [Copy][constructor])
```

6.13.2.6 TEST_CASE() [6/9]

```
TEST_CASE (
            "Testuojamas kopijavimo priskyrimo operatorius" ,
            "" [Copy][assignment][operator])
```

6.13.2.7 TEST_CASE() [7/9]

```
TEST_CASE (
            "Testuojamas move konstruktorius" ,
            "" [Move][constructor])
```

6.13.2.8 TEST_CASE() [8/9]

```
TEST_CASE (
            "Testuojamas move priskyrimo operatorius" ,
            "" [Move][assignment][operator])
```

6.13.2.9 TEST_CASE() [9/9]

```
TEST_CASE (
            "Testuojami getteriai" ,
            "" [Getter])
```

Index