v2.0

Karina Stūronaitė

Generated by Doxygen 1.12.0

1 Hierarchical Index 1

1.1 Class Hierarchy . . . 1

2 Class Index 3

2.1 Class List . . . 3

3 File Index 5

3.1 File List . . . 5

4 Class Documentation 7

4.1 Studentas Class Reference . . . 7

4.1.1 Detailed Description . . . 8

4.1.2 Constructor & Destructor Documentation . . . 8

4.1.2.1 Studentas() [1/3] . . . 8

4.1.2.2 Studentas() [2/3] . . . 9

4.1.2.3 Studentas() [3/3] . . . 9

4.1.2.4 ~Studentas() . . . 9

4.1.3 Member Function Documentation . . . 9

4.1.3.1 addNamuDarbai() . . . 9

4.1.3.2 generuotiPazymius() . . . 9

4.1.3.3 operator=() . . . 10

4.1.3.4 printInfo() . . . 10

4.1.3.5 setNamuDarbai() . . . 10

4.1.3.6 skaiciuotiGalutiniMediana() . . . 10

4.1.3.7 skaiciuotiGalutiniVidurki() . . . 11

4.1.4 Friends And Related Symbol Documentation . . . 11

4.1.4.1 operator<< . . . 11

4.1.4.2 operator>> . . . 11

4.2 Zmogus Class Reference . . . 12

4.2.1 Detailed Description . . . 12

4.2.2 Constructor & Destructor Documentation . . . 13

4.2.2.1 Zmogus() [1/2] . . . 13

4.2.2.2 Zmogus() [2/2] . . . 13

4.2.2.3 ~Zmogus() . . . 13

4.2.3 Member Function Documentation . . . 13

4.2.3.1 getPavarde() . . . 13

4.2.3.2 getVardas() . . . 14

4.2.3.3 printInfo() . . . 14

4.2.3.4 setPavarde() . . . 14

4.2.3.5 setVardas() . . . 14

4.2.4 Member Data Documentation . . . 14

4.2.4.1 pavarde . . . 14

4.2.4.2 vardas . . . 14

5 File Documentation **15**

5.1 funkcijos.cpp File Reference . . . 15

5.1.1 Function Documentation . . . 16

5.1.1.1 generuotiAtsitiktiniPazymi() . . . 16

5.1.1.2 generuotiPazymius() . . . 17

5.1.1.3 generuotiStudentuFailus() . . . 17

5.1.1.4 isaugotiStudentuGrupe() [1/2] . . . 17

5.1.1.5 isaugotiStudentuGrupe() [2/2] . . . 18

5.1.1.6 ivestiStudentuDuomenis() [1/2] . . . 18

5.1.1.7 ivestiStudentuDuomenis() [2/2] . . . 18

5.1.1.8 nuskaitytiStudentus() [1/2] . . . 19

5.1.1.9 nuskaitytiStudentus() [2/2] . . . 19

5.1.1.10 readInteger() . . . 20

5.1.1.11 skaiciuotiGalutiniMediana() [1/2] . . . 20

5.1.1.12 skaiciuotiGalutiniMediana() [2/2] . . . 20

5.1.1.13 skaiciuotiGalutiniVidurki() [1/2] . . . 20

5.1.1.14 skaiciuotiGalutiniVidurki() [2/2] . . . 21

5.1.1.15 strategija1() [1/2] . . . 21

5.1.1.16 strategija1() [2/2] . . . 22

5.1.1.17 strategija2() [1/2] . . . 22

5.1.1.18 strategija2() [2/2] . . . 22

5.1.1.19 strategija3() [1/2] . . . 23

5.1.1.20 strategija3() [2/2] . . . 23

5.1.1.21 testKonteinerius() [1/2] . . . 24

5.1.1.22 testKonteinerius() [2/2] . . . 24

5.1.1.23 trysMetodai() . . . 25

5.2 funkcijos.h File Reference . . . 25

5.2.1 Detailed Description . . . 27

5.2.2 Function Documentation . . . 27

5.2.2.1 generuotiPazymius() . . . 27

5.2.2.2 generuotiStudentuFailus() . . . 27

5.2.2.3 isaugotiStudentuGrupe() [1/2] . . . 27

5.2.2.4 isaugotiStudentuGrupe() [2/2] . . . 28

5.2.2.5 ivestiStudentuDuomenis() [1/2] . . . 28

5.2.2.6 ivestiStudentuDuomenis() [2/2] . . . 29

5.2.2.7 nuskaitytiStudentus() [1/2] . . . 29

5.2.2.8 nuskaitytiStudentus() [2/2] . . . 30

5.2.2.9 readInteger() . . . 30

5.2.2.10 strategija1() [1/2] . . . 31

5.2.2.11 strategija1() [2/2] . . . 31

5.2.2.12 strategija2() [1/2] . . . 32

5.2.2.13 strategija2() [2/2] . . . 32

5.2.2.14 strategija3() [1/2] . . . 33
5.2.2.15 strategija3() [2/2] . . . 34
5.2.2.16 testKonteinerius() [1/2] . . . 34
5.2.2.17 testKonteinerius() [2/2] . . . 35
5.2.2.18 trysMetodai() . . . 36
5.3 funkcijos.h . . . 36
5.4 main.cpp File Reference . . . 37
5.4.1 Function Documentation . . . 38
5.4.1.1 main() . . . 38
5.5 studentas.h File Reference . . . 38
5.5.1 Detailed Description . . . 39
5.6 studentas.h . . . 39
5.7 test.cpp File Reference . . . 40
5.7.1 Function Documentation . . . 41
5.7.1.1 TEST() [1/3] . . . 41
5.7.1.2 TEST() [2/3] . . . 41
5.7.1.3 TEST() [3/3] . . . 41
5.8 testavimas/testavimas.cpp File Reference . . . 41
5.8.1 Function Documentation . . . 41
5.8.1.1 main() . . . 41
5.8.1.2 TEST() [1/5] . . . 42
5.8.1.3 TEST() [2/5] . . . 42
5.8.1.4 TEST() [3/5] . . . 42
5.8.1.5 TEST() [4/5] . . . 42
5.8.1.6 TEST() [5/5] . . . 42
5.9 zmogus.h File Reference . . . 42
5.9.1 Detailed Description . . . 43
5.10 zmogus.h . . . 43

Index **45**

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Zmogus 12
  Studentas 7

Chapter 2

Class Index

2.1 Class List

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

Studentas
: Klase studentui aprasyti 7

Zmogus
: Abstrakti baze klase zmogui aprasyti 12

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

funkcijos.cpp 15
funkcijos.h
    Apraso naudojamas funkcijas 25
main.cpp 37
studentas.h
    Apraso isvestine klase `Studentas` 38
test.cpp 40
zmogus.h
    Apraso abstrakcia baze klase `Zmogus` 42
testavimas/testavimas.cpp 41

Chapter 4

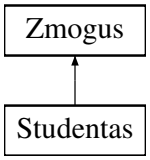
Class Documentation

4.1 Studentas Class Reference

Klase studentui aprasyti.

`#include <studentas.h>`

Inheritance diagram for Studentas:



Public Member Functions

- Studentas ()
  Numatytojo konstruktoriaus deklaracija.
- Studentas (const std::string &v, const std::string &p, const std::vector< int > &nd, int egz)
  Konstruktorius su parametrais.
- Studentas (const Studentas &other)
  Kopijavimo konstruktorius.
- ~Studentas ()
  Destruktorius.
- Studentas & operator= (const Studentas &other)
  Priskyrimo operatorius.
- void setNamuDarbai (const std::vector< int > &nd)
  Nustato studento namu darbu pazymius.
- void addNamuDarbai (int pazymys)
  Prideda viena namu darbu pazymi.
- void generuotiPazymius (int kiek)
  Sugeneruoja atsitiktinius namu darbu pazymius.
- double skaiciuotiGalutiniVidurki () const
  Skaiciuoja galutini ivertinima kaip vidurki.
- double skaiciuotiGalutiniMediana () const
  Skaiciuoja galutini ivertinima kaip mediana.
- virtual void printInfo () const override
  Isveda studento informacija.

Public Member Functions inherited from Zmogus

- Zmogus ()

  Numatytojo konstruktoriaus deklaracija.
- Zmogus (const std::string &v, const std::string &p)

  Konstruktorius su parametrais.
- virtual ~Zmogus ()

  Virtualus destruktorius.
- virtual std::string getVardas () const

  Grazina zmogaus varda.
- virtual void setVardas (const std::string &v)

  Nustato zmogaus varda.
- virtual std::string getPavarde () const

  Grazina zmogaus pavarde.
- virtual void setPavarde (const std::string &p)

  Nustato zmogaus pavarde.

Friends

- std::ostream & operator<< (std::ostream &out, const Studentas &s)

  Isveda studento informacija i srauta.
- std::istream & operator>> (std::istream &is, Studentas &studentas)

  Nuskaitymo operatorius.

Additional Inherited Members

Protected Attributes inherited from Zmogus

- std::string vardas

  Zmogaus vardas.
- std::string pavarde

  Zmogaus pavarde.

4.1.1 Detailed Description

Klase studentui aprasyti.

`Studentas` yra isvestine klase is `Zmogus`. Ji prideda savybes ir metodus, skirtus dirbti su studento namu darbu rezultatais ir egzamino pazymiu.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Studentas() [1/3]

```
Studentas::Studentas ()  [inline]
```

Numatytojo konstruktoriaus deklaracija.

Inicializuoja varda, pavarde kaip tuscias eilutes ir egzamino pazymi kaip 0.

4.1.2.2 Studentas() [2/3]

```
Studentas::Studentas (
            const std::string & v,
            const std::string & p,
            const std::vector< int > & nd,
            int egz)  [inline]
```

Konstruktorius su parametrais.

Parameters

v	Vardas.
p	Pavarde.
nd	Namu darbu pazymiai.
egz	Egzamino pazymys.

4.1.2.3 Studentas() [3/3]

```
Studentas::Studentas (
            const Studentas & other)  [inline]
```

Kopijavimo konstruktorius.

Parameters

other	Kitas `Studentas` objektas, kuris bus nukopijuotas.

4.1.2.4 ~Studentas()

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

Destruktorius.

4.1.3 Member Function Documentation

4.1.3.1 addNamuDarbai()

```
void Studentas::addNamuDarbai (
            int pazymys)  [inline]
```

Prideda viena namu darbu pazymi.

Parameters

pazymys	Naujas pazymys.

4.1.3.2 generuotiPazymius()

```
void Studentas::generuotiPazymius (
            int kiek)  [inline]
```

Sugeneruoja atsitiktinius namu darbu pazymius.

Parameters

kiek	Kiek pazymiu sugeneruoti.

4.1.3.3 operator=()

```
Studentas & Studentas::operator= (
            const Studentas & other)  [inline]
```

Priskyrimo operatorius.

Parameters

other	Kitas `Studentas` objektas, kuris bus priskirtas.

Returns

Nuoroda i priskirta objekta.

4.1.3.4 printInfo()

```
virtual void Studentas::printInfo () const  [inline], [override], [virtual]
```

Isveda studento informacija.

Implements Zmogus.

4.1.3.5 setNamuDarbai()

```
void Studentas::setNamuDarbai (
            const std::vector< int > & nd)  [inline]
```

Nustato studento namu darbu pazymius.

Parameters

nd	Nauji namu darbu pazymiai.

4.1.3.6 skaiciuotiGalutiniMediana()

```
double Studentas::skaiciuotiGalutiniMediana () const  [inline]
```

Skaiciuoja galutini ivertinima kaip mediana.

Returns

Galutinis ivertinimas.

4.1.3.7 skaiciuotiGalutiniVidurki()

```
double Studentas::skaiciuotiGalutiniVidurki () const  [inline]
```

Skaiciuoja galutini ivertinima kaip vidurki.

Returns

Galutinis ivertinimas.

4.1.4 Friends And Related Symbol Documentation

4.1.4.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Studentas & s)  [friend]
```

Isveda studento informacija i srauta.

Parameters

out	Isvedimo srautas.
s	Studento objektas.

Returns

Atnaujintas isvedimo srautas.

4.1.4.2 operator>>

```
std::istream & operator>> (
            std::istream & is,
            Studentas & studentas)  [friend]
```

Nuskaitymo operatorius.

Parameters

is	Nuskaitymo srautas.
studentas	Studento objektas.

Returns

Atnaujintas nuskaitymo srautas.

Nuskaitymo operatorius, skirtas isvedimui. Leidžia vartotojui įvesti studento varda, pavarde, namu darbu pazymius arba generuoti juos atsitiktinai ir ivesti egzamino pazymi.

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

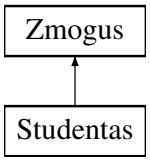
- studentas.h

4.2 Zmogus Class Reference

Abstrakti baze klase zmogui aprasyti.

`#include <zmogus.h>`

Inheritance diagram for Zmogus:



Public Member Functions

- Zmogus ()
  Numatytojo konstruktoriaus deklaracija.
- Zmogus (const std::string &v, const std::string &p)
  Konstruktorius su parametrais.
- virtual ~Zmogus ()
  Virtualus destruktorius.
- virtual void printInfo () const =0
  Grynai virtuali funkcija, skirta informacijai isvesti.
- virtual std::string getVardas () const
  Grazina zmogaus varda.
- virtual void setVardas (const std::string &v)
  Nustato zmogaus varda.
- virtual std::string getPavarde () const
  Grazina zmogaus pavarde.
- virtual void setPavarde (const std::string &p)
  Nustato zmogaus pavarde.

Protected Attributes

- std::string vardas
  Zmogaus vardas.
- std::string pavarde
  Zmogaus pavarde.

4.2.1 Detailed Description

Abstrakti baze klase zmogui aprasyti.

Klase `Zmogus` apibrezia bendrus atributus ir funkcionaluma, kuris yra paveldimas isvestinese klaseje. Ji yra abstrakti, todel negalima sukurti `Zmogus` klase objektu tiesiogiai.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Zmogus() [1/2]

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

Numatytojo konstruktoriaus deklaracija.

Inicializuoja varda ir pavarde kaip tuscias eilutes.

4.2.2.2 Zmogus() [2/2]

```
Zmogus::Zmogus (
            const std::string & v,
            const std::string & p)  [inline]
```

Konstruktorius su parametrais.

Parameters

v	Vardas.
p	Pavarde.

Inicializuoja `vardas` ir `pavarde` nurodytomis reiksmes.

4.2.2.3 ~Zmogus()

```
virtual Zmogus::~Zmogus ()  [inline], [virtual]
```

Virtualus destruktorius.

Uztikrina teisinga paveldetos klases istekliu valyma.

4.2.3 Member Function Documentation

4.2.3.1 getPavarde()

```
virtual std::string Zmogus::getPavarde () const  [inline], [virtual]
```

Grazina zmogaus pavarde.

Returns

Zmogaus pavarde.

4.2.3.2 getVardas()

```
virtual std::string Zmogus::getVardas () const  [inline], [virtual]
```

Grazina zmogaus varda.

Returns

Zmogaus vardas.

4.2.3.3 printInfo()

```
virtual void Zmogus::printInfo () const  [pure virtual]
```

Grynai virtuali funkcija, skirta informacijai isvesti.

Si funkcija turi buti igyvendinta isvestinese klaseje.

Implemented in Studentas.

4.2.3.4 setPavarde()

```
virtual void Zmogus::setPavarde (
            const std::string & p)  [inline], [virtual]
```

Nustato zmogaus pavarde.

Parameters

p	Naujaja pavarde.

4.2.3.5 setVardas()

```
virtual void Zmogus::setVardas (
            const std::string & v)  [inline], [virtual]
```

Nustato zmogaus varda.

Parameters

v	Naujasis vardas.

4.2.4 Member Data Documentation

4.2.4.1 pavarde

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

Zmogaus pavarde.

4.2.4.2 vardas

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

Zmogaus vardas.

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

- zmogus.h

Chapter 5

File Documentation

5.1 funkcijos.cpp File Reference

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

Functions

- int readInteger ()

  Funkcija, skirta ivesti sveikaji skaiciu su patikra, kad jis butu teigiamas.
- void generuotiStudentuFailus ()

  Funkcija, skirta generuoti studentu failus su atsitiktiniais duomenimis.
- int generuotiAtsitiktiniPazymi (int min=1, int max=10)

  Funkcija, skirta generuoti atsitiktini pazymi.
- void generuotiPazymius (Studentas &studentas, int namuDarbuKiekis)

  Funkcija, skirta generuoti studento pazymius.
- double skaiciuotiGalutiniVidurki (const std::vector< int > &namuDarbai, int egzaminas)

  Funkcija, skirta skaiciuoti galutini studento vidurki.
- double skaiciuotiGalutiniMediana (const std::vector< int > &namuDarbai)

  Funkcija, skirta skaiciuoti galutine mediana.
- double skaiciuotiGalutiniVidurki (const std::list< int > &namuDarbai, int egzaminas)

  Funkcija, skirta skaiciuoti galutini studento vidurki su lista.
- double skaiciuotiGalutiniMediana (const std::list< int > &namuDarbai)

  Funkcija, skirta skaiciuoti galutine mediana su lista.
- void ivestiStudentuDuomenis (std::vector< Studentas > &studentaiVektorius)

  Funkcija, skirta ivesti studentu duomenis i vektoriu.
- void ivestiStudentuDuomenis (std::list< Studentas > &studentaiSarasas)

  Funkcija, skirta ivesti studentu duomenis i sarasa.
- void nuskaitytiStudentus (const std::string &failoPavadinimas, std::vector< Studentas > &studentai)

  Nuskaityti studentus is failo ir issaugoti i vektoriu.
- void nuskaitytiStudentus (const std::string &failoPavadinimas, std::list< Studentas > &studentai)

  Nuskaityti studentus is failo ir issaugoti i sarasa.
- void isaugotiStudentuGrupe (const std::vector< Studentas > &studentai)

  Iraso studentu grupe i ekrana.

- void isaugotiStudentuGrupe (const std::list< Studentas > &studentai)

  Iraso studentu grupe i ekrana.
- void strategija1 (std::vector< Studentas > &studentai, bool pagalVidurki, std::vector< Studentas > &vargsiukai, std::vector< Studentas > &kietekai)

  Strategija 1: Studentu suskirstymas i vargsiukus ir kietekus naudojant viena kontineri.
- void strategija1 (std::list< Studentas > &studentai, bool pagalVidurki, std::list< Studentas > &vargsiukai, std::list< Studentas > &kietekai)

  Strategija 1: Studentu suskirstymas i vargsiukus ir kietekus naudojant viena konteineri.
- void strategija2 (std::vector< Studentas > &studentai, bool pagalVidurki, std::vector< Studentas > &vargsiukai, std::vector< Studentas > &kietekai)

  Strategija 2: Studentu suskirstymas i vargsiukus ir kietekus naudojant 2 atskirus konteinerius.
- void strategija2 (std::list< Studentas > &studentai, bool pagalVidurki, std::list< Studentas > &vargsiukai, std::list< Studentas > &kietekai)

  Studentu suskirstymas i vargsiukus ir kietekus naudojant 2 atskirus konteinerius.
- void strategija3 (std::vector< Studentas > &studentai, bool pagalVidurki, std::vector< Studentas > &vargsiukai, std::vector< Studentas > &kietekai)

  Strategija 3: Studentu suskirstymas i vargsiukus ir kietekus naudojant geriausia strategija.
- void strategija3 (std::list< Studentas > &studentai, bool pagalVidurki, std::list< Studentas > &vargsiukai, std::list< Studentas > &kietekai)

  Strategija 3: Studentu suskirstymas i vargsiukus ir kietekus naudojant geriausia strategija.
- void testKonteinerius (const std::vector< Studentas > &studentaiVector, bool pagalVidurki, double &totalPartitionTimeVec, double &totalNuskriaustukaiSaveTimeVec, double &totalKietiakaiSaveTimeVec, void(∗strategija)(std::vector< Studentas > &, bool, std::vector< Studentas > &, std::vector< Studentas > &))

  Testuoja konteinerius su std::vector<Studentas>.
- void testKonteinerius (const std::list< Studentas > &studentaiList, bool pagalVidurki, double &totalPartitionTimeList, double &totalNuskriaustukaiSaveTimeList, double &totalKietiakaiSaveTimeList, void(∗strategija)(std::list< Studentas > &, bool, std::list< Studentas > &, std::list< Studentas > &))

  Testuoja konteinerius su std::list<Studentas>.
- void trysMetodai ()

  Demonstracija su trimis studento objektais.

5.1.1 Function Documentation

5.1.1.1 generuotiAtsitiktiniPazymi()

```
int generuotiAtsitiktiniPazymi (
            int min = 1,
            int max = 10)
```

Funkcija, skirta generuoti atsitiktini pazymi.

Funkcija sugeneruoja atsitiktini pazymi pagal nurodytus minimalu ir maksimalu diapazonus.

Parameters

min	Minimalus galimas pazymys (numatytoji reiksme: 1).
max	Maksimalus galimas pazymys (numatytoji reiksme: 10).

Returns

int Atsitiktinai sugeneruotas pazymys.

5.1.1.2 generuotiPazymius()

```
void generuotiPazymius (
            Studentas & studentas,
            int namuDarbuKiekis)
```

Funkcija, skirta generuoti studento pazymius.

Generuoti pazymius studentui.

Si funkcija sugeneruoja atsitiktinius pazymius studentui uz nurodyta namu darbu kieki.

Parameters

studentas	Studentas, kuriam generuojami pazymiai.
namuDarbuKiekis	Namu darbu kieki, pagal kuri bus sugeneruoti pazymiai.

Returns

void

5.1.1.3 generuotiStudentuFailus()

```
void generuotiStudentuFailus ()
```

Funkcija, skirta generuoti studentu failus su atsitiktiniais duomenimis.

Generuoti studentu failus.

Si funkcija sukuria kelis studentu duomenu failus su atsitiktiniais pazymiais uz namu darbus ir egzaminus. Failai kuriami pagal is anksto nustatyta irasu kieki (1000, 10000 ir t.t.). Jeigu failas jau egzistuoja, jis bus praleistas.

Returns

void

5.1.1.4 isaugotiStudentuGrupe() [1/2]

```
void isaugotiStudentuGrupe (
            const std::list< Studentas > & studentai)
```

Iraso studentu grupe i ekrana.

Isaugoti studentu grupe i faila (naudojant sarasus)

Si funkcija parodo studentu duomenis (vardas, pavarde, galutinis vidurkis ir galutinis mediana) ekrane is pateikto studentu saraso.

Parameters

studentai	Sarasas, kuriame yra studentai, kuriu duomenys bus atvaizduoti.

5.1.1.5 isaugotiStudentuGrupe() [2/2]

```
void isaugotiStudentuGrupe (
            const std::vector< Studentas > & studentai)
```

Iraso studentu grupe i ekrana.

Isaugoti studentu grupe i faila (naudojant vektorius)

Si funkcija parodo studentu duomenis (vardas, pavarde, galutinis vidurkis ir galutinis mediana) ekrane is pateikto studentu vektoriaus.

Parameters

studentai	Vektorius, kuriame yra studentai, kuriu duomenys bus atvaizduoti.

5.1.1.6 ivestiStudentuDuomenis() [1/2]

```
void ivestiStudentuDuomenis (
            std::list< Studentas > & studentaiSarasas)
```

Funkcija, skirta ivesti studentu duomenis i sarasa.

Ivesti studentu duomenis i sarasa.

Si funkcija leidzia vartotojui ivesti studentu duomenis ir saugoti juos sarase.

Parameters

studentaiSarasas	Sarasas, kuriame bus saugomi studentai.

Returns

void

5.1.1.7 ivestiStudentuDuomenis() [2/2]

```
void ivestiStudentuDuomenis (
            std::vector< Studentas > & studentaiVektorius)
```

Funkcija, skirta ivesti studentu duomenis i vektoriu.

Ivesti studentu duomenis i vektoriu.

Si funkcija leidzia vartotojui ivesti studentu duomenis ir saugoti juos vektoriuje.

Parameters

studentaiVektorius	Vektorius, kuriame bus saugomi studentai.

Returns

void

5.1.1.8 nuskaitytiStudentus() [1/2]

```
void nuskaitytiStudentus (
            const std::string & failoPavadinimas,
            std::list< Studentas > & studentai)
```

Nuskaityti studentus is failo ir issaugoti i sarasa.

Nuskaityti studentus is failo i sarasa.

Si funkcija atidaro faila pagal pateikta pavadinima, nuskaityti studento duomenis (vardas, pavarde, namu darbai ir egzaminas) ir issaugo juos i sarasa.

Parameters

failoPavadinimas	Failo, is kurio bus nuskaityti studentu duomenys, pavadinimas.
studentai	Sarasas, kuriame bus issaugoti studentai.

5.1.1.9 nuskaitytiStudentus() [2/2]

```
void nuskaitytiStudentus (
            const std::string & failoPavadinimas,
            std::vector< Studentas > & studentai)
```

Nuskaityti studentus is failo ir issaugoti i vektoriu.

Nuskaityti studentus is failo i vektoriu.

Si funkcija atidaro faila pagal pateikta pavadinima, nuskaityti studento duomenis (vardas, pavarde, namu darbai ir egzaminas) ir issaugo juos i vektoriu.

Parameters

failoPavadinimas	Failo, is kurio bus nuskaityti studentu duomenys, pavadinimas.
studentai	Vektorius, kuriame bus issaugoti studentai.

5.1.1.10 readInteger()

```
int readInteger ()
```

Funkcija, skirta ivesti sveikaji skaiciu su patikra, kad jis butu teigiamas.

Nuskaityti sveika skaiciu nuo vartotojo.

Si funkcija nuolat praso vartotojo ivesti teigiamaji sveikaji skaiciu. Jeigu ivestas skaicius yra klaidingas arba neigiamas, bus rodomas klaidos pranesimas ir vartotojas bus paprasytas ivesti skaiciu is naujo, kol bus ivestas teisingas teigiamas skaicius.

Returns

int Teisingai ivestas sveikasis skaicius.

5.1.1.11 skaiciuotiGalutiniMediana() [1/2]

```
double skaiciuotiGalutiniMediana (
            const std::list< int > & namuDarbai)
```

Funkcija, skirta skaiciuoti galutine mediana su lista.

Si funkcija apskaiciuoja galutine studento pazymiu mediana is visu namu darbu.

Parameters

namuDarbai	Studentui uzduoti namu darbu pazymiai.

Returns

double Galutine mediana.

5.1.1.12 skaiciuotiGalutiniMediana() [2/2]

```
double skaiciuotiGalutiniMediana (
            const std::vector< int > & namuDarbai)
```

Funkcija, skirta skaiciuoti galutine mediana.

Si funkcija apskaiciuoja galutine studento pazymiu mediana is visu namu darbu.

Parameters

namuDarbai	Studentui uzduoti namu darbu pazymiai.

Returns

double Galutine mediana.

5.1.1.13 skaiciuotiGalutiniVidurki() [1/2]

```
double skaiciuotiGalutiniVidurki (
            const std::list< int > & namuDarbai,
            int egzaminas)
```

Funkcija, skirta skaiciuoti galutini studento vidurki su lista.

Si funkcija apskaiciuoja galutini studento vidurki, atsizvelgiant i visus namu darbus ir egzamina.

Parameters

namuDarbai	Studentui uzduoti namu darbu pazymiai.
egzaminas	Egzamino pazymys.

Returns

double Galutinis vidurkis.

5.1.1.14 skaiciuotiGalutiniVidurki() [2/2]

```
double skaiciuotiGalutiniVidurki (
            const std::vector< int > & namuDarbai,
            int egzaminas)
```

Funkcija, skirta skaiciuoti galutini studento vidurki.

Si funkcija apskaiciuoja galutini studento vidurki, atsizvelgiant i visus namu darbus ir egzamina.

Parameters

namuDarbai	Studentui uzduoti namu darbu pazymiai.
egzaminas	Egzamino pazymys.

Returns

double Galutinis vidurkis.

5.1.1.15 strategija1() [1/2]

```
void strategija1 (
            std::list< Studentas > & studentai,
            bool pagalVidurki,
            std::list< Studentas > & vargsiukai,
            std::list< Studentas > & kietekai)
```

Strategija 1: Studentu suskirstymas i vargsiukus ir kietekus naudojant viena konteineri.

Pirmoji strategija studentu rusiavimui (naudoja sarasa)

Si funkcija suskirsto studentus i dvi grupes - vargsiukus (studentai su galutiniu rezultatu maziau nei 5) ir kietekus (studentai su galutiniu rezultatu 5 ir daugiau) pagal vidurki arba mediana. Naudoja sarasa vietoj vektoriaus.

Parameters

studentai	Sarasas studentu, kurie bus suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Sarasas, i kuri bus irasomi vargsiukai.
kietekai	Sarasas, i kuri bus irasomi kietekai.

5.1.1.16 strategija1() [2/2]

```
void strategija1 (
            std::vector< Studentas > & studentai,
            bool pagalVidurki,
            std::vector< Studentas > & vargsiukai,
            std::vector< Studentas > & kietekai)
```

Strategija 1: Studentu suskirstymas i vargsiukus ir kietekus naudojant viena kontineri.

Pirmoji strategija studentu rusiavimui (naudoja vektorius)

Si funkcija suskirsto studentus i dvi grupes - vargsiukus (studentai su galutiniu rezultatu maziau nei 5) ir kietekus (studentai su galutiniu rezultatu 5 ir daugiau) pagal vidurki arba mediana.

Parameters

studentai	Vektorius studentu, kurie bus suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Vektorius, i kuri bus irasomi vargsiukai.
kietekai	Vektorius, i kuri bus irasomi kietekai.

5.1.1.17 strategija2() [1/2]

```
void strategija2 (
            std::list< Studentas > & studentai,
            bool pagalVidurki,
            std::list< Studentas > & vargsiukai,
            std::list< Studentas > & kietekai)
```

Studentu suskirstymas i vargsiukus ir kietekus naudojant 2 atskirus konteinerius.

Antroji strategija studentu rusiavimui (naudoja sarasus)

Si funkcija naudoja sarasa ir perkelia studentus i atskiras grupes (vargsiukai ir kietekai), remiantis galutiniu rezultatu, kuris apskaiciuojamas pagal vidurki arba medianna. Studentai su galutiniu rezultatu maziau nei 5 yra perkelti i vargsiuku grupe, o visi kiti - i kieteku grupe.

Parameters

studentai	Sarasas studentu, kuriuos reikia suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Sarasas, kuriame bus issaugoti vargsiukai.
kietekai	Sarasas, kuriame bus issaugoti kietekai.

5.1.1.18 strategija2() [2/2]

```
void strategija2 (
            std::vector< Studentas > & studentai,
            bool pagalVidurki,
            std::vector< Studentas > & vargsiukai,
            std::vector< Studentas > & kietekai)
```

Strategija 2: Studentu suskirstymas i vargsiukus ir kietekus naudojant 2 atskirus konteinerius.

Antroji strategija studentu rusiavimui (naudoja vektorius)

Si funkcija naudoja vektoriu ir perkelia studentus i atskiras grupes (vargsiukai ir kietekai), remiantis galutiniu rezultatu, kuris apskaiciuojamas pagal vidurki arba medianna. Studentai su galutiniu rezultatu maziau nei 5 yra perkelti i vargsiuku grupe, o visi kiti - i kieteku grupe.

Parameters

studentai	Vektorius studentu, kuriuos reikia suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Vektorius, kuriame bus issaugoti vargsiukai.
kietekai	Vektorius, kuriame bus issaugoti kietekai.

5.1.1.19 strategija3() [1/2]

```
void strategija3 (
            std::list< Studentas > & studentai,
            bool pagalVidurki,
            std::list< Studentas > & vargsiukai,
            std::list< Studentas > & kietekai)
```

Strategija 3: Studentu suskirstymas i vargsiukus ir kietekus naudojant geriausia strategija.

Trecioji strategija studentu rusiavimui (naudoja sarasus)

Si funkcija naudoja sarasa ir padalina studentus i vargsiukus ir kietekus. Studentai, kuriu galutinis rezultatas yra maziau nei 5, bus perkelti i vargsiuku grupe, o visi kiti - i kieteku grupe.

Parameters

studentai	Sarasas studentu, kuriuos reikia suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Sarasas, kuriame bus issaugoti vargsiukai.
kietekai	Sarasas, kuriame bus issaugoti kietekai.

5.1.1.20 strategija3() [2/2]

```
void strategija3 (
            std::vector< Studentas > & studentai,
            bool pagalVidurki,
            std::vector< Studentas > & vargsiukai,
            std::vector< Studentas > & kietekai)
```

Strategija 3: Studentu suskirstymas i vargsiukus ir kietekus naudojant geriausia strategija.

Trecioji strategija studentu rusiavimui (naudoja vektorius)

Si funkcija naudoja vektoriu ir padalina studentus i vargsiukus ir kietekus, naudodama std::partition funkcija. Studentai, kuriu galutinis rezultatas yra maziau nei 5, bus perkelti i vargsiuku grupe, o visi kiti - i kieteku grupe.

Parameters

studentai	Vektorius studentu, kuriuos reikia suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Vektorius, kuriame bus issaugoti vargsiukai.
kietekai	Vektorius, kuriame bus issaugoti kietekai.

5.1.1.21 testKonteinerius() [1/2]

```
void testKonteinerius (
            const std::list< Studentas > & studentaiList,
            bool pagalVidurki,
            double & totalPartitionTimeList,
            double & totalNuskriaustukaiSaveTimeList,
            double & totalKietiakaiSaveTimeList,
            void(* strategija )(std::list< Studentas > &, bool, std::list< Studentas > &,
std::list< Studentas > &))
```

Testuoja konteinerius su std::list<Studentas>.

Testuoti konteinerius su duomenimis.

Funkcija testuoja strategija su std::list<Studentas> konteineriu, suskirstydama studentus i vargsiukus ir kietekus. Atlikus strategija, rezultatai gali buti isvedami i ekrana arba issaugomi faile. Matuojamas laikas, reikalingas grupei suskirstyti ir rezultatams issaugoti.

Parameters

studentaiList	Sarasas studentu, kurie bus testuojami.
pagalVidurki	Kintamasis, nurodantis, ar naudoti vidurki (true) ar mediana (false) kaip galutini rezultata.
totalPartitionTimeList	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo suskirstyti studentai.
totalNuskriaustukaiSaveTimeList	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo issaugoti nuskriaustukai.
totalKietiakaiSaveTimeList	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo issaugoti kietekai.
strategija	Funkcija, kuri atlieka studentu suskirstyma pagal tam tikra strategija.

5.1.1.22 testKonteinerius() [2/2]

```
void testKonteinerius (
            const std::vector< Studentas > & studentaiVector,
            bool pagalVidurki,
            double & totalPartitionTimeVec,
            double & totalNuskriaustukaiSaveTimeVec,
            double & totalKietiakaiSaveTimeVec,
            void(* strategija )(std::vector< Studentas > &, bool, std::vector< Studentas >
&, std::vector< Studentas > &))
```

Testuoja konteinerius su std::vector<Studentas>.

Testuoti konteinerius su duomenimis.

Funkcija testuoja strategija su std::vector<Studentas> konteineriu, suskirstydama studentus i vargsiukus ir kietekus. Atlikus strategija, rezultatai gali buti isvedami i ekrana arba issaugomi faile. Matuojamas laikas, reikalingas grupei suskirstyti ir rezultatams issaugoti.

Parameters

studentaiVector	Vektorius studentu, kurie bus testuojami.

Parameters

pagalVidurki	Kintamasis, nurodantis, ar naudoti vidurki (true) ar mediana (false) kaip galutini rezultata.
totalPartitionTimeVec	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo suskirstyti studentai.
totalNuskriaustukaiSaveTimeVec	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo issaugoti nuskriaustukai.
totalKietiakaiSaveTimeVec	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo issaugoti kietekai.
strategija	Funkcija, kuri atlieka studentu suskirstyma pagal tam tikra strategija.

5.1.1.23 trysMetodai()

```
void trysMetodai ()
```

Demonstracija su trimis studento objektais.

Triju metodu testavimas.

Funkcija demonstruoja, kaip veikia priskyrimo operatorius ir kopijavimo konstruktorius su Studentas objektais.

Note

Funkcija sukuria studento objekta `a`, tada priskiria ji i objekta `b` ir sukuria objekta `c` naudodama kopijavimo konstruktoriumi.

Parameters

none	

5.2 funkcijos.h File Reference

Apraso naudojamas funkcijas.

```
#include <iostream>
#include <iomanip>
#include <vector>
#include <list>
#include <string>
#include <fstream>
#include <sstream>
#include <random>
#include <algorithm>
#include <numeric>
#include <functional>
#include <iterator>
#include <chrono>
#include <limits>
```

Functions

- int readInteger ()

  Nuskaityti sveika skaiciu nuo vartotojo.
- void generuotiStudentuFailus ()

  Generuoti studentu failus.
- void generuotiPazymius (Studentas &studentas, int namuDarbuKiekis)

  Generuoti pazymius studentui.
- void nuskaitytiStudentus (const std::string &failoPavadinimas, std::vector< Studentas > &studentai)

  Nuskaityti studentus is failo i vektoriu.
- void nuskaitytiStudentus (const std::string &failoPavadinimas, std::list< Studentas > &studentai)

  Nuskaityti studentus is failo i sarasa.
- void strategija1 (std::vector< Studentas > &studentai, bool pagalVidurki, std::vector< Studentas > &vargsiukai, std::vector< Studentas > &kietekai)

  Pirmoji strategija studentu rusiavimui (naudoja vektorius)
- void strategija1 (std::list< Studentas > &studentai, bool pagalVidurki, std::list< Studentas > &vargsiukai, std::list< Studentas > &kietekai)

  Pirmoji strategija studentu rusiavimui (naudoja sarasa)
- void strategija2 (std::vector< Studentas > &studentai, bool pagalVidurki, std::vector< Studentas > &vargsiukai, std::vector< Studentas > &kietekai)

  Antroji strategija studentu rusiavimui (naudoja vektorius)
- void strategija2 (std::list< Studentas > &studentai, bool pagalVidurki, std::list< Studentas > &vargsiukai, std::list< Studentas > &kietekai)

  Antroji strategija studentu rusiavimui (naudoja sarasus)
- void strategija3 (std::vector< Studentas > &studentai, bool pagalVidurki, std::vector< Studentas > &vargsiukai, std::vector< Studentas > &kietekai)

  Trecioji strategija studentu rusiavimui (naudoja vektorius)
- void strategija3 (std::list< Studentas > &studentai, bool pagalVidurki, std::list< Studentas > &vargsiukai, std::list< Studentas > &kietekai)

  Trecioji strategija studentu rusiavimui (naudoja sarasus)
- void isaugotiStudentuGrupe (const std::vector< Studentas > &studentai)

  Isaugoti studentu grupe i faila (naudojant vektorius)
- void isaugotiStudentuGrupe (const std::list< Studentas > &studentai)

  Isaugoti studentu grupe i faila (naudojant sarasus)
- void ivestiStudentuDuomenis (std::vector< Studentas > &studentaiVektorius)

  Ivesti studentu duomenis i vektoriu.
- void ivestiStudentuDuomenis (std::list< Studentas > &studentaiSarasas)

  Ivesti studentu duomenis i sarasa.
- void testKonteinerius (const std::vector< Studentas > &studentaiVektorius, bool pagalVidurki, double &totalPartitionTimeVec, double &totalNuskriaustukaiSaveTimeVec, double &totalKietiakaiSaveTimeVec, void(*strategija)(std::vector< Studentas > &, bool, std::vector< Studentas > &, std::vector< Studentas > &))

  Testuoti konteinerius su duomenimis.
- void testKonteinerius (const std::list< Studentas > &studentaiList, bool pagalVidurki, double &totalPartitionTimeList, double &totalNuskriaustukaiSaveTimeList, double &totalKietiakaiSaveTimeList, void(*strategija)(std::list< Studentas > &, bool, std::list< Studentas > &, std::list< Studentas > &))

  Testuoti konteinerius su duomenimis.
- void trysMetodai ()

  Triju metodu testavimas.

5.2.1 Detailed Description

Apraso naudojamas funkcijas.

Siame faile pateikiamos funkcijos.

5.2.2 Function Documentation

5.2.2.1 generuotiPazymius()

```
void generuotiPazymius (
            Studentas & studentas,
            int namuDarbuKiekis)
```

Generuoti pazymius studentui.

Funkcija sugeneruoja atsitiktinius pazymius studentui.

Parameters

studentas	Studentas, kuriam bus sugeneruoti pazymiai
namuDarbuKiekis	Namu darbu kiekis, kuriems bus sugeneruoti pazymiai

Generuoti pazymius studentui.

Si funkcija sugeneruoja atsitiktinius pazymius studentui uz nurodyta namu darbu kieki.

Parameters

studentas	Studentas, kuriam generuojami pazymiai.
namuDarbuKiekis	Namu darbu kieki, pagal kuri bus sugeneruoti pazymiai.

Returns

void

5.2.2.2 generuotiStudentuFailus()

```
void generuotiStudentuFailus ()
```

Generuoti studentu failus.

Funkcija generuoja studentu duomenis ir isaugo juos failuose.

Generuoti studentu failus.

Si funkcija sukuria kelis studentu duomenu failus su atsitiktiniais pazymiais uz namu darbus ir egzaminus. Failai kuriami pagal is anksto nustatyta irasu kieki (1000, 10000 ir t.t.). Jeigu failas jau egzistuoja, jis bus praleistas.

Returns

void

5.2.2.3 isaugotiStudentuGrupe() [1/2]

```
void isaugotiStudentuGrupe (
            const std::list< Studentas > & studentai)
```

Isaugoti studentu grupe i faila (naudojant sarasus)

Funkcija isaugoti studentu grupe i faila naudojant sarasus.

Parameters

studentai	Studentai, kuriuos norima issaugoti

Isaugoti studentu grupe i faila (naudojant sarasus)

Si funkcija parodo studentu duomenis (vardas, pavarde, galutinis vidurkis ir galutinis mediana) ekrane is pateikto studentu saraso.

Parameters

studentai	Sarasas, kuriame yra studentai, kuriu duomenys bus atvaizduoti.

5.2.2.4 isaugotiStudentuGrupe() [2/2]

```
void isaugotiStudentuGrupe (
            const std::vector< Studentas > & studentai)
```

Isaugoti studentu grupe i faila (naudojant vektorius)

Funkcija isaugoti studentu grupe i faila naudojant vektorius.

Parameters

studentai	Studentai, kuriuos norima issaugoti

Isaugoti studentu grupe i faila (naudojant vektorius)

Si funkcija parodo studentu duomenis (vardas, pavarde, galutinis vidurkis ir galutinis mediana) ekrane is pateikto studentu vektoriaus.

Parameters

studentai	Vektorius, kuriame yra studentai, kuriu duomenys bus atvaizduoti.

5.2.2.5 ivestiStudentuDuomenis() [1/2]

```
void ivestiStudentuDuomenis (
            std::list< Studentas > & studentaiSarasas)
```

Ivesti studentu duomenis i sarasa.

Funkcija leidzia ivesti studentu duomenis ir issaugoti juos sarase.

Parameters

studentaiSarasas	Sarasas, kuriame bus issaugoti studentu duomenys

Ivesti studentu duomenis i sarasa.

Si funkcija leidzia vartotojui ivesti studentu duomenis ir saugoti juos sarase.

Parameters

studentaiSarasas	Sarasas, kuriame bus saugomi studentai.

Returns

void

5.2.2.6 ivestiStudentuDuomenis() [2/2]

```
void ivestiStudentuDuomenis (
            std::vector< Studentas > & studentaiVektorius)
```

Ivesti studentu duomenis i vektoriu.

Funkcija leidzia ivesti studentu duomenis ir issaugoti juos vektoriuje.

Parameters

studentaiVektorius	Vektorius, kuriame bus issaugoti studentu duomenys

Ivesti studentu duomenis i vektoriu.

Si funkcija leidzia vartotojui ivesti studentu duomenis ir saugoti juos vektoriuje.

Parameters

studentaiVektorius	Vektorius, kuriame bus saugomi studentai.

Returns

void

5.2.2.7 nuskaitytiStudentus() [1/2]

```
void nuskaitytiStudentus (
            const std::string & failoPavadinimas,
            std::list< Studentas > & studentai)
```

Nuskaityti studentus is failo i sarasa.

Funkcija nuskaityti studentu duomenis is failo ir juos issaugoti sarase.

Parameters

failoPavadinimas	Failo pavadinimas is kurio bus nuskaityti studentai
studentai	Sarasas, kuriame bus issaugoti studentai

Nuskaityti studentus is failo i sarasa.

Si funkcija atidaro faila pagal pateikta pavadinima, nuskaityti studento duomenis (vardas, pavarde, namu darbai ir egzaminas) ir issaugo juos i sarasa.

Parameters

failoPavadinimas	Failo, is kurio bus nuskaityti studentu duomenys, pavadinimas.
studentai	Sarasas, kuriame bus issaugoti studentai.

5.2.2.8 nuskaitytiStudentus() [2/2]

```
void nuskaitytiStudentus (
            const std::string & failoPavadinimas,
            std::vector< Studentas > & studentai)
```

Nuskaityti studentus is failo i vektoriu.

Funkcija nuskaityti studentu duomenis is failo ir juos issaugoti vektoriuje.

Parameters

failoPavadinimas	Failo pavadinimas is kurio bus nuskaityti studentai
studentai	Vektorius, kuriame bus issaugoti studentai

Nuskaityti studentus is failo i vektoriu.

Si funkcija atidaro faila pagal pateikta pavadinima, nuskaityti studento duomenis (vardas, pavarde, namu darbai ir egzaminas) ir issaugo juos i vektoriu.

Parameters

failoPavadinimas	Failo, is kurio bus nuskaityti studentu duomenys, pavadinimas.
studentai	Vektorius, kuriame bus issaugoti studentai.

5.2.2.9 readInteger()

```
int readInteger ()
```

Nuskaityti sveika skaiciu nuo vartotojo.

Funkcija atlieka sveiko skaiciaus nuskaityma is vartotojo.

Returns

Nuskaitytas sveikas skaicius

Nuskaityti sveika skaiciu nuo vartotojo.

Si funkcija nuolat praso vartotojo ivesti teigiamaji sveikaji skaiciu. Jeigu ivestas skaicius yra klaidingas arba neigiamas, bus rodomas klaidos pranesimas ir vartotojas bus paprasytas ivesti skaiciu is naujo, kol bus ivestas teisingas teigiamas skaicius.

Returns

int Teisingai ivestas sveikasis skaicius.

5.2.2.10 strategija1() [1/2]

```
void strategija1 (
            std::list< Studentas > & studentai,
            bool pagalVidurki,
            std::list< Studentas > & vargsiukai,
            std::list< Studentas > & kietekai)
```

Pirmoji strategija studentu rusiavimui (naudoja sarasa)

Funkcija, atliekanti pirma strategija studentu padalinimui i dvi grupes (vargsiukai ir kietekai) pagal pasirinktus kriterijus.

Parameters

studentai	Studentai, kurie bus apdoroti
pagalVidurki	Nustato, ar rusiavimas bus pagal vidurki
vargsiukai	Grupes, kuriuose bus issaugoti vargsesni studentai
kietekai	Grupes, kuriuose bus issaugoti geresni studentai

Pirmoji strategija studentu rusiavimui (naudoja sarasa)

Si funkcija suskirsto studentus i dvi grupes - vargsiukus (studentai su galutiniu rezultatu maziau nei 5) ir kietekus (studentai su galutiniu rezultatu 5 ir daugiau) pagal vidurki arba mediana. Naudoja sarasa vietoj vektoriaus.

Parameters

studentai	Sarasas studentu, kurie bus suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Sarasas, i kuri bus irasomi vargsiukai.
kietekai	Sarasas, i kuri bus irasomi kietekai.

5.2.2.11 strategija1() [2/2]

```
void strategija1 (
            std::vector< Studentas > & studentai,
            bool pagalVidurki,
            std::vector< Studentas > & vargsiukai,
            std::vector< Studentas > & kietekai)
```

Pirmoji strategija studentu rusiavimui (naudoja vektorius)

Funkcija, atliekanti pirma strategija studentu padalinimui i dvi grupes (vargsiukai ir kietekai) pagal pasirinktus kriterijus.

Parameters

studentai	Studentai, kurie bus apdoroti
pagalVidurki	Nustato, ar rusiavimas bus pagal vidurki
vargsiukai	Grupes, kuriuose bus issaugoti vargsesni studentai
kietekai	Grupes, kuriuose bus issaugoti geresni studentai

Pirmoji strategija studentu rusiavimui (naudoja vektorius)

Si funkcija suskirsto studentus i dvi grupes - vargsiukus (studentai su galutiniu rezultatu maziau nei 5) ir kietekus (studentai su galutiniu rezultatu 5 ir daugiau) pagal vidurki arba mediana.

Parameters

studentai	Vektorius studentu, kurie bus suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Vektorius, i kuri bus irasomi vargsiukai.
kietekai	Vektorius, i kuri bus irasomi kietekai.

5.2.2.12 strategija2() [1/2]

```
void strategija2 (
            std::list< Studentas > & studentai,
            bool pagalVidurki,
            std::list< Studentas > & vargsiukai,
            std::list< Studentas > & kietekai)
```

Antroji strategija studentu rusiavimui (naudoja sarasus)

Funkcija, atliekanti antraja strategija studentu padalinimui i dvi grupes (vargsiukai ir kietekai) pagal pasirinktus kriterijus.

Parameters

studentai	Studentai, kurie bus apdoroti
pagalVidurki	Nustato, ar rusiavimas bus pagal vidurki
vargsiukai	Grupes, kuriuose bus issaugoti vargsesni studentai
kietekai	Grupes, kuriuose bus issaugoti geresni studentai

Antroji strategija studentu rusiavimui (naudoja sarasus)

Si funkcija naudoja sarasa ir perkelia studentus i atskiras grupes (vargsiukai ir kietekai), remiantis galutiniu rezultatu, kuris apskaiciuojamas pagal vidurki arba medianna. Studentai su galutiniu rezultatu maziau nei 5 yra perkelti i vargsiuku grupe, o visi kiti - i kieteku grupe.

Parameters

studentai	Sarasas studentu, kuriuos reikia suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Sarasas, kuriame bus issaugoti vargsiukai.
kietekai	Sarasas, kuriame bus issaugoti kietekai.

5.2.2.13 strategija2() [2/2]

```
void strategija2 (
            std::vector< Studentas > & studentai,
            bool pagalVidurki,
            std::vector< Studentas > & vargsiukai,
            std::vector< Studentas > & kietekai)
```

Antroji strategija studentu rusiavimui (naudoja vektorius)

Funkcija, atliekanti antraja strategija studentu padalinimui i dvi grupes (vargsiukai ir kietekai) pagal pasirinktus kriterijus.

Parameters

studentai	Studentai, kurie bus apdoroti
pagalVidurki	Nustato, ar rusiavimas bus pagal vidurki
vargsiukai	Grupes, kuriuose bus issaugoti vargsesni studentai
kietekai	Grupes, kuriuose bus issaugoti geresni studentai

Antroji strategija studentu rusiavimui (naudoja vektorius)

Si funkcija naudoja vektoriu ir perkelia studentus i atskiras grupes (vargsiukai ir kietekai), remiantis galutiniu rezultatu, kuris apskaiciuojamas pagal vidurki arba medianna. Studentai su galutiniu rezultatu maziau nei 5 yra perkelti i vargsiuku grupe, o visi kiti - i kieteku grupe.

Parameters

studentai	Vektorius studentu, kuriuos reikia suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Vektorius, kuriame bus issaugoti vargsiukai.
kietekai	Vektorius, kuriame bus issaugoti kietekai.

5.2.2.14 strategija3() [1/2]

```
void strategija3 (
            std::list< Studentas > & studentai,
            bool pagalVidurki,
            std::list< Studentas > & vargsiukai,
            std::list< Studentas > & kietekai)
```

Trecioji strategija studentu rusiavimui (naudoja sarasus)

Funkcija, atliekanti treciaja strategija studentu padalinimui i dvi grupes (vargsiukai ir kietekai) pagal pasirinktus kriterijus.

Parameters

studentai	Studentai, kurie bus apdoroti
pagalVidurki	Nustato, ar rusiavimas bus pagal vidurki
vargsiukai	Grupes, kuriuose bus issaugoti vargsesni studentai
kietekai	Grupes, kuriuose bus issaugoti geresni studentai

Trecioji strategija studentu rusiavimui (naudoja sarasus)

Si funkcija naudoja sarasa ir padalina studentus i vargsiukus ir kietekus. Studentai, kuriu galutinis rezultatas yra maziau nei 5, bus perkelti i vargsiuku grupe, o visi kiti - i kieteku grupe.

Parameters

studentai	Sarasas studentu, kuriuos reikia suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Sarasas, kuriame bus issaugoti vargsiukai.
kietekai	Sarasas, kuriame bus issaugoti kietekai.

5.2.2.15 strategija3() [2/2]

```
void strategija3 (
            std::vector< Studentas > & studentai,
            bool pagalVidurki,
            std::vector< Studentas > & vargsiukai,
            std::vector< Studentas > & kietekai)
```

Trecioji strategija studentu rusiavimui (naudoja vektorius)

Funkcija, atliekanti treciaja strategija studentu padalinimui i dvi grupes (vargsiukai ir kietekai) pagal pasirinktus kriterijus.

Parameters

studentai	Studentai, kurie bus apdoroti
pagalVidurki	Nustato, ar rusiavimas bus pagal vidurki
vargsiukai	Grupes, kuriuose bus issaugoti vargsesni studentai
kietekai	Grupes, kuriuose bus issaugoti geresni studentai

Trecioji strategija studentu rusiavimui (naudoja vektorius)

Si funkcija naudoja vektoriu ir padalina studentus i vargsiukus ir kietekus, naudodama std::partition funkcija. Studentai, kuriu galutinis rezultatas yra maziau nei 5, bus perkelti i vargsiuku grupe, o visi kiti - i kieteku grupe.

Parameters

studentai	Vektorius studentu, kuriuos reikia suskirstyti.
pagalVidurki	Jei true, naudojamas galutinis vidurkis, jei false - galutinis mediana.
vargsiukai	Vektorius, kuriame bus issaugoti vargsiukai.
kietekai	Vektorius, kuriame bus issaugoti kietekai.

5.2.2.16 testKonteinerius() [1/2]

```
void testKonteinerius (
            const std::list< Studentas > & studentaiList,
            bool pagalVidurki,
            double & totalPartitionTimeList,
            double & totalNuskriaustukaiSaveTimeList,
            double & totalKietiakaiSaveTimeList,
            void(* strategija )(std::list< Studentas > &, bool, std::list< Studentas > &,
std::list< Studentas > &))
```

Testuoti konteinerius su duomenimis.

Funkcija atlieka konteineriu testavimus su studentu duomenimis naudojant pasirinktus rusiavimo metodus.

Parameters

studentaiList	Studentai, kurie bus naudojami testavimui
pagalVidurki	Nustato, ar rusiavimas bus pagal vidurki
totalPartitionTimeList	Laikas partition funkcijai

totalNuskriaustukaiSaveTimeList	Laikas isaugoti vargsesnius studentus
totalKietiakaiSaveTimeList	Laikas isaugoti kietesnius studentus
strategija	Funkcija, kuri apdoroja studentus pagal pasirinkta strategija

Testuoti konteinerius su duomenimis.

Funkcija testuoja strategija su std::list<Studentas> konteineriu, suskirstydama studentus i vargsiukus ir kietekus. Atlikus strategija, rezultatai gali buti isvedami i ekrana arba issaugomi faile. Matuojamas laikas, reikalingas grupei suskirstyti ir rezultatams issaugoti.

Parameters

studentaiList	Sarasas studentu, kurie bus testuojami.
pagalVidurki	Kintamasis, nurodantis, ar naudoti vidurki (true) ar mediana (false) kaip galutini rezultata.
totalPartitionTimeList	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo suskirstyti studentai.
totalNuskriaustukaiSaveTimeList	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo issaugoti nuskriaustukai.
totalKietiakaiSaveTimeList	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo issaugoti kietekai.
strategija	Funkcija, kuri atlieka studentu suskirstyma pagal tam tikra strategija.

5.2.2.17 testKonteinerius() [2/2]

```
void testKonteinerius (
            const std::vector< Studentas > & studentaiVector,
            bool pagalVidurki,
            double & totalPartitionTimeVec,
            double & totalNuskriaustukaiSaveTimeVec,
            double & totalKietiakaiSaveTimeVec,
            void(* strategija )(std::vector< Studentas > &, bool, std::vector< Studentas >
&, std::vector< Studentas > &))
```

Testuoti konteinerius su duomenimis.

Funkcija atlieka konteineriu testavimus su studentu duomenimis naudojant pasirinktus rusiavimo metodus.

Parameters

studentaiVektorius	Studentai, kurie bus naudojami testavimui
pagalVidurki	Nustato, ar rusiavimas bus pagal vidurki
totalPartitionTimeVec	Laikas partition funkcijai
totalNuskriaustukaiSaveTimeVec	Laikas isaugoti vargsesnius studentus
totalKietiakaiSaveTimeVec	Laikas isaugoti kietesnius studentus
strategija	Funkcija, kuri apdoroja studentus pagal pasirinkta strategija

Testuoti konteinerius su duomenimis.

Funkcija testuoja strategija su std::vector<Studentas> konteineriu, suskirstydama studentus i vargsiukus ir kietekus. Atlikus strategija, rezultatai gali buti isvedami i ekrana arba issaugomi faile. Matuojamas laikas, reikalingas grupei suskirstyti ir rezultatams issaugoti.

Parameters

studentaiVector	Vektorius studentu, kurie bus testuojami.
pagalVidurki	Kintamasis, nurodantis, ar naudoti vidurki (true) ar mediana (false) kaip galutini rezultata.
totalPartitionTimeVec	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo suskirstyti studentai.
totalNuskriaustukaiSaveTimeVec	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo issaugoti nuskriaustukai.
totalKietiakaiSaveTimeVec	Busimas kintamasis, kuriame bus saugomas laikas, per kuri buvo issaugoti kietekai.
strategija	Funkcija, kuri atlieka studentu suskirstyma pagal tam tikra strategija.

5.2.2.18 trysMetodai()

```
void trysMetodai ()
```

Triju metodu testavimas.

Funkcija atlieka triju metodu testavimus.

Triju metodu testavimas.

Funkcija demonstruoja, kaip veikia priskyrimo operatorius ir kopijavimo konstruktorius su Studentas objektais.

Note

Funkcija sukuria studento objekta `a`, tada priskiria ji i objekta `b` ir sukuria objekta `c` naudodama kopijavimo konstruktoriumi.

Parameters

none	

5.3 funkcijos.h

Go to the documentation of this file.

```
00001
00007 #ifndef FUNKCIJOS_H
00008 #define FUNKCIJOS_H
00009
00010 #include <iostream>
00011 #include <iomanip>
00012 #include <vector>
00013 #include <list>
00014 #include <string>
00015 #include <fstream>
00016 #include <sstream>
00017 #include <random>
00018 #include <algorithm>
00019 #include <numeric>
00020 #include <functional>
00021 #include <iterator>
00022 #include <chrono>
00023 #include <limits>
00024
00025 class Studentas;
```

```
00026
00034 int readInteger();
00035
00041 void generuotiStudentuFailus();
00042
00051 void generuotiPazymius(Studentas& studentas, int namuDarbuKiekis);
00052
00061 void nuskaitytiStudentus(const std::string& failoPavadinimas, std::vector<Studentas>& studentai);
00062
00071 void nuskaitytiStudentus(const std::string& failoPavadinimas, std::list<Studentas>& studentai);
00072
00083 void strategija1(std::vector<Studentas>& studentai, bool pagalVidurki, std::vector<Studentas>&
      vargsiukai, std::vector<Studentas>& kietekai);
00084
00095 void strategija1(std::list<Studentas>& studentai, bool pagalVidurki, std::list<Studentas>& vargsiukai,
      std::list<Studentas>& kietekai);
00096
00107 void strategija2(std::vector<Studentas>& studentai, bool pagalVidurki, std::vector<Studentas>&
      vargsiukai, std::vector<Studentas>& kietekai);
00108
00119 void strategija2(std::list<Studentas>& studentai, bool pagalVidurki, std::list<Studentas>& vargsiukai,
      std::list<Studentas>& kietekai);
00120
00131 void strategija3(std::vector<Studentas>& studentai, bool pagalVidurki, std::vector<Studentas>&
      vargsiukai, std::vector<Studentas>& kietekai);
00132
00143 void strategija3(std::list<Studentas>& studentai, bool pagalVidurki, std::list<Studentas>& vargsiukai,
      std::list<Studentas>& kietekai);
00144
00152 void isaugotiStudentuGrupe(const std::vector<Studentas>& studentai);
00153
00161 void isaugotiStudentuGrupe(const std::list<Studentas>& studentai);
00162
00170 void ivestiStudentuDuomenis(std::vector<Studentas>& studentaiVektorius);
00171
00179 void ivestiStudentuDuomenis(std::list<Studentas>& studentaiSarasas);
00180
00193 void testKonteinerius(const std::vector<Studentas>& studentaiVektorius, bool pagalVidurki,
00194     double& totalPartitionTimeVec, double& totalNuskriaustukaiSaveTimeVec, double&
      totalKietiakaiSaveTimeVec,
00195     void (*strategija)(std::vector<Studentas>&, bool, std::vector<Studentas>&,
      std::vector<Studentas>&));
00196
00209 void testKonteinerius(const std::list<Studentas>& studentaiList, bool pagalVidurki,
00210     double& totalPartitionTimeList, double& totalNuskriaustukaiSaveTimeList, double&
      totalKietiakaiSaveTimeList,
00211     void (*strategija)(std::list<Studentas>&, bool, std::list<Studentas>&, std::list<Studentas>&));
00212
00218 void trysMetodai();
00219
00220 #endif // FUNKCIJOS_H
```

5.4 main.cpp File Reference

```
#include "funkcijos.h"
#include "studentas.h"
#include <functional>
#include <iostream>
#include <chrono>
#include <vector>
#include <list>
```

Functions

- int main ()

5.4.1 Function Documentation

5.4.1.1 main()

```
int main ()
```

Tikrina, ar vartotojas nori sugeneruoti studentu failus.

Jei vartotojas pasirenka "y" arba "Y", iskvieciama funkcija generuotiStudentuFailus.

Pasirinkimas, jei vartotojas nusprendzia naudoti vector konteineri.

Jei pasirenkamas vector konteineris, priklausomai nuo vartotojo pasirinkimo, arba ivesti studentu duomenis ranka, arba nuskaityti juos is failo. Po to pasirenkama ir pritaikoma norima strategija.

Pasirenkama ir pritaikoma strategija pagal vartotojo pasirinkima.

Naudojama funkcija testKonteinerius, kad atlikti reikiamus veiksmus su vektoriu.

Pasirinkimas, jei vartotojas nusprendzia naudoti list konteineri.

Jei pasirenkamas list konteineris, priklausomai nuo vartotojo pasirinkimo, arba ivesti studentu duomenis ranka, arba nuskaityti juos is failo. Po to pasirenkama ir pritaikoma norima strategija.

Pasirenkama ir pritaikoma strategija pagal vartotojo pasirinkima.

Naudojama funkcija testKonteinerius, kad atlikti reikiamus veiksmus su list.

Testuoja tris metodus.

Tikrinamas triju metodu veikimas.

Sukuria studento objekta ir isveda jo informacija.

Sukuriamas studento objektas su vardo, pavardes, pazymiu ir amziaus duomenimis, po to isvedama visa informacija apie si studenta.

5.5 studentas.h File Reference

Apraso isvestine klase Studentas.

```
#include "zmogus.h"
#include <vector>
#include <numeric>
#include <algorithm>
#include <iomanip>
```

Classes

- class Studentas

  Klase studentui aprasyti.

5.5.1 Detailed Description

Apraso isvestine klase `Studentas`.

Siame faile pateikiama `Studentas` klase, paveldeta is `Zmogus`. Ji apraso studento duomenis, iskaitant namu darbus ir egzamino rezultatus.

5.6 studentas.h

Go to the documentation of this file.

```
00001
00009 #ifndef STUDENTAS_H
00010 #define STUDENTAS_H
00011
00012 #include "zmogus.h"
00013 #include <vector>
00014 #include <numeric>
00015 #include <algorithm>
00016 #include <iomanip>
00017
00025 class Studentas : public Zmogus {
00026 private:
00027     std::vector<int> namuDarbai;
00028     int egzaminas;
00029
00030 public:
00036     Studentas() : Zmogus(), egzaminas(0) {}
00037
00045     Studentas(const std::string& v, const std::string& p, const std::vector<int>& nd, int egz)
00046         : Zmogus(v, p), namuDarbai(nd), egzaminas(egz) {}
00047
00052     Studentas(const Studentas& other)
00053         : Zmogus(other.vardas, other.pavarde), namuDarbai(other.namuDarbai),
      egzaminas(other.egzaminas) {}
00054
00058     ~Studentas() {}
00059
00065     Studentas& operator=(const Studentas& other) {
00066         if (this != &other) {
00067             vardas = other.vardas;
00068             pavarde = other.pavarde;
00069             namuDarbai = other.namuDarbai;
00070             egzaminas = other.egzaminas;
00071         }
00072         return *this;
00073     }
00074
00079     void setNamuDarbai(const std::vector<int>& nd) {
00080         namuDarbai = nd;
00081     }
00082
00087     void addNamuDarbai(int pazymys) {
00088         namuDarbai.push_back(pazymys);
00089     }
00090
00095     void generuotiPazymius(int kiek) {
00096         namuDarbai.clear();
00097         for (int i = 0; i < kiek; ++i) {
00098             namuDarbai.push_back(1 + rand() % 10);
00099         }
00100     }
00101
00106     double skaiciuotiGalutiniVidurki() const {
00107         if (namuDarbai.empty()) return egzaminas;
00108         double total = std::accumulate(namuDarbai.begin(), namuDarbai.end(), 0);
00109         return (total + egzaminas) / (namuDarbai.size() + 1);
00110     }
00111
00116     double skaiciuotiGalutiniMediana() const {
00117         if (namuDarbai.empty()) return egzaminas;
00118         std::vector<int> sortedNamuDarbai = namuDarbai;
00119         std::sort(sortedNamuDarbai.begin(), sortedNamuDarbai.end());
00120         size_t size = sortedNamuDarbai.size();
00121         return (size % 2 == 0) ?
00122             (sortedNamuDarbai[size / 2 - 1] + sortedNamuDarbai[size / 2]) / 2.0
00123             : sortedNamuDarbai[size / 2];
00124     }
00125
```

```
00132     friend std::ostream& operator«(std::ostream& out, const Studentas& s) {
00133         out « std::left « std::setw(20) « s.vardas
00134             « std::setw(20) « s.pavarde
00135             « std::setw(20) « std::fixed « std::setprecision(2) « s.skaiciuotiGalutiniVidurki()
00136             « std::fixed « std::setprecision(2) « s.skaiciuotiGalutiniMediana() « std::endl;
00137         return out;
00138     }
00139
00151     friend std::istream& operator»(std::istream& is, Studentas& studentas) {
00152         std::cout « "Iveskite studento varda: ";
00153         std::getline(is » std::ws, studentas.vardas);
00154
00155         std::cout « "Iveskite studento pavarde: ";
00156         std::getline(is » std::ws, studentas.pavarde);
00157
00158         std::cout « "Norite:\n";
00159         std::cout « "1 - Ivesti namu darbu pazymius ranka\n";
00160         std::cout « "2 - Sugeneruoti pazymius atsitiktinai\n";
00161         int pasirinkimas;
00162         do {
00163             std::cout « "Pasirinkimas (1 arba 2): ";
00164             is » pasirinkimas;
00165             if (pasirinkimas != 1 && pasirinkimas != 2) {
00166                 std::cout « "Neteisingas pasirinkimas. Bandykite dar karta.\n";
00167             }
00168         } while (pasirinkimas != 1 && pasirinkimas != 2);
00169
00170         if (pasirinkimas == 1) {
00171             studentas.namuDarbai.clear();
00172             std::cout « "Iveskite namu darbu pazymius (iveskite 0, kai baigsite):\n";
00173             int pazymys;
00174             while (true) {
00175                 std::cout « "Pazymys: ";
00176                 is » pazymys;
00177                 if (pazymys == 0) break;
00178                 if (pazymys < 1 || pazymys > 10) {
00179                     std::cout « "Iveskite pazymi nuo 1 iki 10.\n";
00180                     continue;
00181                 }
00182                 studentas.namuDarbai.push_back(pazymys);
00183             }
00184         }
00185         else if (pasirinkimas == 2) {
00186             std::cout « "Kiek namu darbu pazymiu sugeneruoti? ";
00187             int kiek;
00188             is » kiek;
00189             studentas.generuotiPazymius(kiek);
00190         }
00191
00192         std::cout « "Iveskite egzamino pazymi: ";
00193         is » studentas.egzaminas;
00194
00195         is.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
00196         return is;
00197     }
00198
00202     virtual void printInfo() const override {
00203         std::cout « *this;
00204     }
00205 };
00206
00207 #endif // STUDENTAS_H
```

5.7 test.cpp File Reference

```
#include "funkcijos.h"
#include "studentas.h"
#include <gtest/gtest.h>
```

Functions

- TEST (StudentasTest, Vidurkis)
- TEST (StrategijaTest, Strategija1Test)
- TEST (StrategijaTest, Strategija2Test)

5.7.1 Function Documentation

5.7.1.1 TEST() [1/3]

```
TEST (
            StrategijaTest ,
            Strategija1Test )
```

5.7.1.2 TEST() [2/3]

```
TEST (
            StrategijaTest ,
            Strategija2Test )
```

5.7.1.3 TEST() [3/3]

```
TEST (
            StudentasTest ,
            Vidurkis )
```

5.8 testavimas/testavimas.cpp File Reference

```
#include <gtest/gtest.h>
#include "Studentas.h"
#include <sstream>
```

Functions

- TEST (StudentasTest, DefaultConstructorTest)
- TEST (StudentasTest, ParameterizedConstructorTest)
- TEST (StudentasTest, AddNamuDarbaiTest)
- TEST (StudentasTest, GeneruotiPazymiusTest)
- TEST (StudentasTest, OutputStreamOperatorTest)
- int main (int argc, char **argv)

5.8.1 Function Documentation

5.8.1.1 main()

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

5.8.1.2 TEST() [1/5]

```
TEST (
            StudentasTest ,
            AddNamuDarbaiTest )
```

5.8.1.3 TEST() [2/5]

```
TEST (
            StudentasTest ,
            DefaultConstructorTest )
```

5.8.1.4 TEST() [3/5]

```
TEST (
            StudentasTest ,
            GeneruotiPazymiusTest )
```

5.8.1.5 TEST() [4/5]

```
TEST (
            StudentasTest ,
            OutputStreamOperatorTest )
```

5.8.1.6 TEST() [5/5]

```
TEST (
            StudentasTest ,
            ParameterizedConstructorTest )
```

5.9 zmogus.h File Reference

Apraso abstrakcia baze klase Zmogus.

```
#include <string>
```

Classes

- class Zmogus

  Abstrakti baze klase zmogui aprasyti.

5.9.1 Detailed Description

Apraso abstrakcia baze klase `Zmogus`.

Cia aprasoma abstrakti klase `Zmogus`, kuri naudojama paveldejimui ir suteikia bazines savybes bei metodus zmogui aprasyti. Ji yra pagrindas kitoms isvestinems klasems, tokioms kaip `Studentas`.

5.10 zmogus.h

Go to the documentation of this file.

```
00001
00010 #ifndef ZMOGUS_H
00011 #define ZMOGUS_H
00012
00013 #include <string>
00014
00023 class Zmogus {
00024 protected:
00025     std::string vardas;
00026     std::string pavarde;
00027
00028 public:
00034     Zmogus() : vardas(""), pavarde("") {}
00035
00043     Zmogus(const std::string& v, const std::string& p) : vardas(v), pavarde(p) {}
00044
00050     virtual ~Zmogus() {}
00051
00057     virtual void printInfo() const = 0;
00058
00063     virtual std::string getVardas() const { return vardas; }
00064
00069     virtual void setVardas(const std::string& v) { vardas = v; }
00070
00075     virtual std::string getPavarde() const { return pavarde; }
00076
00081     virtual void setPavarde(const std::string& p) { pavarde = p; }
00082 };
00083
00084 #endif // ZMOGUS_H
```

Index

~Studentas
    Studentas, 9
~Zmogus
    Zmogus, 13

addNamuDarbai
    Studentas, 9

funkcijos.cpp, 15
    generuotiAtsitiktiniPazymi, 16
    generuotiPazymius, 16
    generuotiStudentuFailus, 17
    isaugotiStudentuGrupe, 17, 18
    ivestiStudentuDuomenis, 18
    nuskaitytiStudentus, 19
    readInteger, 19
    skaiciuotiGalutiniMediana, 20
    skaiciuotiGalutiniVidurki, 20, 21
    strategija1, 21
    strategija2, 22
    strategija3, 23
    testKonteinerius, 23, 24
    trysMetodai, 25
funkcijos.h, 25
    generuotiPazymius, 27
    generuotiStudentuFailus, 27
    isaugotiStudentuGrupe, 27, 28
    ivestiStudentuDuomenis, 28, 29
    nuskaitytiStudentus, 29, 30
    readInteger, 30
    strategija1, 30, 31
    strategija2, 32
    strategija3, 33
    testKonteinerius, 34, 35
    trysMetodai, 36

generuotiAtsitiktiniPazymi
    funkcijos.cpp, 16
generuotiPazymius
    funkcijos.cpp, 16
    funkcijos.h, 27
    Studentas, 9
generuotiStudentuFailus
    funkcijos.cpp, 17
    funkcijos.h, 27
getPavarde
    Zmogus, 13
getVardas
    Zmogus, 13

isaugotiStudentuGrupe
    funkcijos.cpp, 17, 18
    funkcijos.h, 27, 28
ivestiStudentuDuomenis
    funkcijos.cpp, 18
    funkcijos.h, 28, 29

main
    main.cpp, 38
    testavimas.cpp, 41
main.cpp, 37
    main, 38

nuskaitytiStudentus
    funkcijos.cpp, 19
    funkcijos.h, 29, 30

operator<<
    Studentas, 11
operator>>
    Studentas, 11
operator=
    Studentas, 10

pavarde
    Zmogus, 14
printInfo
    Studentas, 10
    Zmogus, 14

readInteger
    funkcijos.cpp, 19
    funkcijos.h, 30

setNamuDarbai
    Studentas, 10
setPavarde
    Zmogus, 14
setVardas
    Zmogus, 14
skaiciuotiGalutiniMediana
    funkcijos.cpp, 20
    Studentas, 10
skaiciuotiGalutiniVidurki
    funkcijos.cpp, 20, 21
    Studentas, 10
strategija1
    funkcijos.cpp, 21
    funkcijos.h, 30, 31
strategija2
    funkcijos.cpp, 22
    funkcijos.h, 32

strategija3
funkcijos.cpp, 23
funkcijos.h, 33
Studentas, 7
~Studentas, 9
addNamuDarbai, 9
generuotiPazymius, 9
operator<<, 11
operator>>, 11
operator=, 10
printInfo, 10
setNamuDarbai, 10
skaiciuotiGalutiniMediana, 10
skaiciuotiGalutiniVidurki, 10
Studentas, 8, 9
studentas.h, 38

TEST
test.cpp, 41
testavimas.cpp, 41, 42
test.cpp, 40
TEST, 41
testavimas.cpp
main, 41
TEST, 41, 42
testavimas/testavimas.cpp, 41
testKonteinerius
funkcijos.cpp, 23, 24
funkcijos.h, 34, 35
trysMetodai
funkcijos.cpp, 25
funkcijos.h, 36

vardas
Zmogus, 14

Zmogus, 12
~Zmogus, 13
getPavarde, 13
getVardas, 13
pavarde, 14
printInfo, 14
setPavarde, 14
setVardas, 14
vardas, 14
Zmogus, 13
zmogus.h, 42