My Project

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 . . . 9
4.1.2.1 Studentas() [1/3] . . . 9
4.1.2.2 Studentas() [2/3] . . . 9
4.1.2.3 Studentas() [3/3] . . . 9
4.1.2.4 ~Studentas() . . . 10
4.1.3 Member Function Documentation . . . 10
4.1.3.1 getEgzaminas() . . . 10
4.1.3.2 getGalutinisMed() . . . 10
4.1.3.3 getGalutinisVid() . . . 10
4.1.3.4 getNamuDarbai() . . . 11
4.1.3.5 getPavarde() . . . 11
4.1.3.6 getVardas() . . . 11
4.1.3.7 medianosSkaiciavimas() . . . 11
4.1.3.8 operator=() . . . 11
4.1.3.9 setEgzaminas() . . . 12
4.1.3.10 setNamuDarbai() . . . 12
4.1.3.11 setPavarde() . . . 12
4.1.3.12 setVardas() . . . 13
4.1.3.13 vidurkioSkaiciavimas() . . . 13
4.1.4 Friends And Related Symbol Documentation . . . 13
4.1.4.1 operator<< . . . 13
4.1.4.2 operator>> . . . 14
4.2 Zmogus Class Reference . . . 14
4.2.1 Detailed Description . . . 15
4.2.2 Constructor & Destructor Documentation . . . 15
4.2.2.1 Zmogus() [1/2] . . . 15
4.2.2.2 Zmogus() [2/2] . . . 15
4.2.2.3 ~Zmogus() . . . 16
4.2.3 Member Function Documentation . . . 16
4.2.3.1 getPavarde() . . . 16
4.2.3.2 getVardas() . . . 16

4.2.4 Member Data Documentation . . . 17
4.2.4.1 pavarde . . . 17
4.2.4.2 vardas . . . 17

5 File Documentation **19**

5.1 Failas.cpp File Reference . . . 19
5.1.1 Detailed Description . . . 20
5.1.2 Function Documentation . . . 20
5.1.2.1 galvociu_atrinkimas() . . . 20
5.1.2.2 irasymas_i_faila() . . . 20
5.1.2.3 laiko_skaiciavimas() . . . 20
5.1.2.4 laiko_skaiciavimas_failo_generavimas() . . . 21
5.1.2.5 pagrindinio_failo_generavimas() . . . 21
5.1.2.6 rikiavimas_pagal_pavarde_laikas() . . . 21
5.1.2.7 rikiavimas_pagal_pazymius_laikas() . . . 22
5.1.2.8 rikiavimas_pagal_varda_laikas() . . . 22
5.1.2.9 studentu_isskirstymas() . . . 22
5.1.2.10 vargsiuku_atrinkimas() . . . 23
5.2 Failas.h File Reference . . . 23
5.2.1 Detailed Description . . . 24
5.2.2 Function Documentation . . . 24
5.2.2.1 galvociu_atrinkimas() . . . 24
5.2.2.2 irasymas_i_faila() . . . 24
5.2.2.3 laiko_skaiciavimas() . . . 25
5.2.2.4 laiko_skaiciavimas_failo_generavimas() . . . 25
5.2.2.5 pagrindinio_failo_generavimas() . . . 26
5.2.2.6 rikiavimas_pagal_pavarde_laikas() . . . 26
5.2.2.7 rikiavimas_pagal_pazymius_laikas() . . . 27
5.2.2.8 rikiavimas_pagal_varda_laikas() . . . 27
5.2.2.9 studentu_isskirstymas() . . . 28
5.2.2.10 vargsiuku_atrinkimas() . . . 28
5.3 Failas.h . . . 29
5.4 Header.h File Reference . . . 29
5.5 Header.h . . . 30
5.6 main.cpp File Reference . . . 30
5.6.1 Function Documentation . . . 31
5.6.1.1 main() . . . 31
5.7 strategijos.cpp File Reference . . . 31
5.7.1 Detailed Description . . . 31
5.7.2 Function Documentation . . . 31
5.7.2.1 operator==() . . . 31
5.7.2.2 skaidymas_2_strategija_list() . . . 32

5.7.2.3 skaidymas_2_strategija_vector() . . . 32
5.7.2.4 skaidymas_3_strategija_list() . . . 32
5.7.2.5 skaidymas_3_strategija_vector() . . . 33
5.8 strategijos.h File Reference . . . 33
5.8.1 Detailed Description . . . 34
5.8.2 Function Documentation . . . 34
5.8.2.1 operator==() . . . 34
5.8.2.2 skaidymas_2_strategija_list() . . . 34
5.8.2.3 skaidymas_2_strategija_vector() . . . 35
5.8.2.4 skaidymas_3_strategija_list() . . . 35
5.8.2.5 skaidymas_3_strategija_vector() . . . 36
5.9 strategijos.h . . . 36
5.10 Studentas.cpp File Reference . . . 37
5.10.1 Detailed Description . . . 38
5.10.2 Function Documentation . . . 38
5.10.2.1 irasymas() . . . 38
5.10.2.2 irasymas_list() . . . 38
5.10.2.3 isvedimas() . . . 39
5.10.2.4 isvedimas_list() . . . 39
5.10.2.5 isvedimas_su_mediana() . . . 39
5.10.2.6 isvedimas_su_vidurkiu() . . . 39
5.10.2.7 ivedimas() . . . 40
5.10.2.8 ivesties_isvesties_metodu_demonstracija() . . . 40
5.10.2.9 operator<<() . . . 40
5.10.2.10 operator>>() . . . 40
5.10.2.11 palyginti_pavardes() . . . 41
5.10.2.12 palyginti_pazymius() . . . 41
5.10.2.13 palyginti_vardus() . . . 41
5.10.2.14 random_egz() . . . 42
5.10.2.15 random_pazymiai() . . . 42
5.10.2.16 rule_of_three_metodu_demonstracija() . . . 42
5.10.2.17 skaitymas_is_failo() . . . 42
5.10.2.18 valymas() . . . 43
5.11 Studentas.h File Reference . . . 43
5.11.1 Detailed Description . . . 44
5.11.2 Function Documentation . . . 44
5.11.2.1 irasymas() . . . 44
5.11.2.2 irasymas_list() . . . 45
5.11.2.3 isvedimas() . . . 45
5.11.2.4 isvedimas_list() . . . 45
5.11.2.5 isvedimas_su_mediana() . . . 46
5.11.2.6 isvedimas_su_vidurkiu() . . . 46

5.11.2.7 ivedimas() . . . 46
5.11.2.8 ivesties_isvesties_metodu_demonstracija() . . . 47
5.11.2.9 palyginti_pavardes() . . . 47
5.11.2.10 palyginti_pazymius() . . . 47
5.11.2.11 palyginti_vardus() . . . 48
5.11.2.12 random_egz() . . . 49
5.11.2.13 random_pazymiai() . . . 49
5.11.2.14 rule_of_three_metodu_demonstracija() . . . 49
5.11.2.15 skaitymas_is_failo() . . . 49
5.11.2.16 valymas() . . . 50
5.12 Studentas.h . . . 50
5.13 v0_3.cpp File Reference . . . 52
5.13.1 Function Documentation . . . 52
5.13.1.1 galvociu_atrinkimas_naudojant_list() . . . 52
5.13.1.2 irasymas_naudojant_list() . . . 53
5.13.1.3 laiko_skaiciavimas_list_konteineris() . . . 53
5.13.1.4 rikiavimas_pagal_pavarde_laikas_list() . . . 53
5.13.1.5 rikiavimas_pagal_pazymius_laikas_list() . . . 54
5.13.1.6 rikiavimas_pagal_varda_laikas_list() . . . 54
5.13.1.7 skaitymas_is_failo_list() . . . 54
5.13.1.8 studentu_isskirstymas_list() . . . 55
5.13.1.9 vargsiuku_atrinkimas_naudojant_list() . . . 55
5.14 v0_3_header.h File Reference . . . 56
5.14.1 Detailed Description . . . 56
5.14.2 Function Documentation . . . 56
5.14.2.1 galvociu_atrinkimas_naudojant_list() . . . 56
5.14.2.2 irasymas() . . . 57
5.14.2.3 irasymas_naudojant_list() . . . 57
5.14.2.4 laiko_skaiciavimas_list_konteineris() . . . 58
5.14.2.5 rikiavimas_pagal_pavarde_laikas_list() . . . 58
5.14.2.6 rikiavimas_pagal_pazymius_laikas_list() . . . 59
5.14.2.7 rikiavimas_pagal_varda_laikas_list() . . . 59
5.14.2.8 skaitymas_is_failo_list() . . . 60
5.14.2.9 studentu_isskirstymas_list() . . . 60
5.14.2.10 vargsiuku_atrinkimas_naudojant_list() . . . 61
5.15 v0_3_header.h . . . 62
5.16 Zmogus.cpp File Reference . . . 62
5.16.1 Detailed Description . . . 62
5.17 Zmogus.h File Reference . . . 62
5.17.1 Detailed Description . . . 63
5.18 Zmogus.h . . . 63

Index 65

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Zmogus 14
  Studentas 7

Chapter 2

Class Index

2.1 Class List

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

Studentas
: Represents a student, inheriting from the Zmogus class 7

Zmogus
: Represents a basic person entity with a name and surname 14

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Failas.cpp
This file contains the implementation of a program for managing and processing student data . 19
Failas.h
Header file containing function declarations for managing student data and performing file operations 23
Header.h 29
main.cpp 30
strategijos.cpp
This file contains functions for processing student data with different strategies for reading, sorting, filtering, and writing to files 31
strategijos.h
This header file defines the function prototypes for various strategies related to processing student data using different containers (vector or list) 33
Studentas.cpp
Implementation of the Studentas class and related functions 37
Studentas.h
Declaration of the `Studentas` class 43
v0_3.cpp 52
v0_3_header.h
Function declarations for processing student data using `std::list` 56
Zmogus.cpp
Implementation of the `Zmogus` class, representing a basic person entity 62
Zmogus.h
Declaration of the `Zmogus` class, a base class for representing a person entity 62

Chapter 4

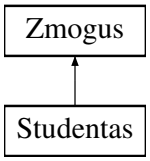
Class Documentation

4.1 Studentas Class Reference

Represents a student, inheriting from the Zmogus class.

`#include <Studentas.h>`

Inheritance diagram for Studentas:



Public Member Functions

- Studentas ()
  Default constructor.
- Studentas (string vardas, string pavarde, vector< int > namu_darbai, int egzaminas)
  Parameterized constructor.
- Studentas (const Studentas &other)
  Copy constructor.
- Studentas & operator= (const Studentas &other)
  Copy assignment operator.
- ~Studentas ()
  Destructor.
- void setVardas (const string &v)
  Sets the student's first name.
- void setPavarde (const std::string &p)
  Sets the student's last name.
- void setNamuDarbai (const std::vector< int > &nd)
  Sets the student's homework grades.
- void setEgzaminas (int egz)
  Sets the student's exam grade.
- const std::string & getVardas () const override

Gets the student's first name.

- const std::string & getPavarde () const override

  Gets the student's last name.
- const std::vector< int > & getNamuDarbai () const

  Gets the student's homework grades.
- int getEgzaminas () const

  Gets the student's exam grade.
- double getGalutinisVid () const

  Gets the student's final grade based on average calculation.
- double getGalutinisMed () const

  Gets the student's final grade based on median calculation.
- void vidurkioSkaiciavimas ()

  Calculates the average grade of the student.
- void medianosSkaiciavimas ()

  Calculates the median grade of the student.

Public Member Functions inherited from Zmogus

- Zmogus ()

  Default constructor.
- Zmogus (string vardas, string pavarde)

  Parameterized constructor.
- virtual ~Zmogus ()=default

  Virtual destructor.

Friends

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

  Output stream operator for Studentas.
- std::istream & operator>> (std::istream &is, Studentas &s)

  Input stream operator for Studentas.

Additional Inherited Members

Protected Attributes inherited from Zmogus

- string vardas

  The first name of the person.
- string pavarde

  The last name of the person.

4.1.1 Detailed Description

Represents a student, inheriting from the Zmogus class.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Studentas() [1/3]

```
Studentas::Studentas ()
```

Default constructor.

Default constructor for Studentas.

4.1.2.2 Studentas() [2/3]

```
Studentas::Studentas (
            string vardas,
            string pavarde,
            vector< int > namu_darbai,
            int egzaminas)
```

Parameterized constructor.

Constructor with parameters for initializing a Studentas object.

Parameters

vardas	Student's first name.
pavarde	Student's last name.
namu_darbai	Homework grades.
egzaminas	Exam grade.
vardas	Student's first name.
pavarde	Student's last name.
namu_darbai	Vector of homework grades.
egzaminas	Exam grade.

4.1.2.3 Studentas() [3/3]

```
Studentas::Studentas (
            const Studentas & other)
```

Copy constructor.

Copy constructor for Studentas.

Parameters

other	Another Studentas object to copy from.
other	The Studentas object to copy from.

4.1.2.4 ~Studentas()

```
Studentas::~Studentas ()
```

Destructor.

Destructor for Studentas.

4.1.3 Member Function Documentation

4.1.3.1 getEgzaminas()

```
int Studentas::getEgzaminas () const
```

Gets the student's exam grade.

Returns

Exam grade.

The exam grade.

4.1.3.2 getGalutinisMed()

```
double Studentas::getGalutinisMed () const
```

Gets the student's final grade based on median calculation.

Gets the student's final grade (median-based).

Returns

Final grade based on median.

The final grade.

4.1.3.3 getGalutinisVid()

```
double Studentas::getGalutinisVid () const
```

Gets the student's final grade based on average calculation.

Gets the student's final grade (average-based).

Returns

Final grade based on average.

The final grade.

4.1.3.4 getNamuDarbai()

```
const std::vector< int > & Studentas::getNamuDarbai () const
```

Gets the student's homework grades.

Returns

Vector of homework grades.

A vector of homework grades.

4.1.3.5 getPavarde()

```
const std::string & Studentas::getPavarde () const  [override], [virtual]
```

Gets the student's last name.

Returns

Last name.

The last name.

Implements Zmogus.

4.1.3.6 getVardas()

```
const std::string & Studentas::getVardas () const  [override], [virtual]
```

Gets the student's first name.

Returns

First name.

The first name.

Implements Zmogus.

4.1.3.7 medianosSkaiciavimas()

```
void Studentas::medianosSkaiciavimas ()
```

Calculates the median grade of the student.

Calculates the student's median grade.

4.1.3.8 operator=()

```
Studentas & Studentas::operator= (
            const Studentas & other)
```

Copy assignment operator.

Copy assignment operator for Studentas.

Parameters

other	Another Studentas object to assign from.

Returns

Reference to the assigned Studentas object.

Parameters

other	The Studentas object to copy from.

Returns

A reference to the modified Studentas object.

4.1.3.9 setEgzaminas()

```
void Studentas::setEgzaminas (
            int egz)
```

Sets the student's exam grade.

Parameters

egz	Exam grade.
egz	The exam grade.

4.1.3.10 setNamuDarbai()

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

Sets the student's homework grades.

Parameters

nd	Vector of homework grades.
nd	A vector of homework grades.

4.1.3.11 setPavarde()

```
void Studentas::setPavarde (
            const std::string & p)
```

Sets the student's last name.

Parameters

p	Last name.
p	The last name.

4.1.3.12 setVardas()

```
void Studentas::setVardas (
            const string & v)
```

Sets the student's first name.

Parameters

v	First name.
v	The first name.

4.1.3.13 vidurkioSkaiciavimas()

```
void Studentas::vidurkioSkaiciavimas ()
```

Calculates the average grade of the student.

Calculates the student's average grade.

4.1.4 Friends And Related Symbol Documentation

4.1.4.1 operator<<

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

Output stream operator for Studentas.

Parameters

os	Output stream.
s	Studentas object.

Returns

Output stream with student information.

Parameters

os	The output stream.
s	The Studentas object to output.

Returns

A reference to the output stream.

4.1.4.2 operator>>

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

Input stream operator for Studentas.

Parameters

is	Input stream.
s	Studentas object.

Returns

Input stream with student information.

Parameters

is	The input stream.
s	The Studentas object to populate.

Returns

A reference to the input stream.

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

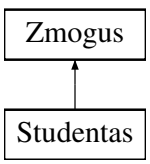
- Studentas.h
- Studentas.cpp

4.2 Zmogus Class Reference

Represents a basic person entity with a name and surname.

```
#include <Zmogus.h>
```

Inheritance diagram for Zmogus:



Public Member Functions

- Zmogus ()
  Default constructor.
- Zmogus (string vardas, string pavarde)
  Parameterized constructor.
- virtual ~Zmogus ()=default
  Virtual destructor.
- virtual const std::string & getVardas () const =0
  Pure virtual getter for the first name.
- virtual const std::string & getPavarde () const =0
  Pure virtual getter for the last name.

Protected Attributes

- string vardas
  The first name of the person.
- string pavarde
  The last name of the person.

4.2.1 Detailed Description

Represents a basic person entity with a name and surname.

The `Zmogus` class provides protected attributes for a person's first name (`vardas`) and last name (`pavarde`). It is designed as a base class with pure virtual methods for accessing these attributes, making it abstract.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Zmogus() [1/2]

```
Zmogus::Zmogus ()
```

Default constructor.

Default constructor for the `Zmogus` class.

Initializes the `Zmogus` object with empty strings for the first name (`vardas`) and last name (`pavarde`).

4.2.2.2 Zmogus() [2/2]

```
Zmogus::Zmogus (
            string vardas,
            string pavarde)
```

Parameterized constructor.

Parameterized constructor for the `Zmogus` class.

Parameters

vardas	The first name of the person.
pavarde	The last name of the person.

Initializes the `Zmogus` object with specified first name (`vardas`) and last name (`pavarde`).

Parameters

vardas	The first name of the person.
pavarde	The last name of the person.

4.2.2.3 ~Zmogus()

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

Virtual destructor.

4.2.3 Member Function Documentation

4.2.3.1 getPavarde()

```
virtual const std::string & Zmogus::getPavarde () const  [pure virtual]
```

Pure virtual getter for the last name.

Returns

A constant reference to the `pavarde` string.

Implemented in Studentas.

4.2.3.2 getVardas()

```
virtual const std::string & Zmogus::getVardas () const  [pure virtual]
```

Pure virtual getter for the first name.

Returns

A constant reference to the `vardas` string.

Implemented in Studentas.

4.2.4 Member Data Documentation

4.2.4.1 pavarde

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

The last name of the person.

4.2.4.2 vardas

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

The first name of the person.

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

- Zmogus.h
- Zmogus.cpp

Chapter 5

File Documentation

5.1 Failas.cpp File Reference

This file contains the implementation of a program for managing and processing student data.

```
#include <stdio.h>
#include "Studentas.h"
#include "Header.h"
#include "Failas.h"
#include "v0_3_header.h"
```

Functions

- void pagrindinio_failo_generavimas (int studentu_skaicius, int nd_skaicius)

  Generates the main file with student data including names, grades, and exam scores.
- void irasymas_i_faila (vector< Studentas > &studentai, string failo_pav)

  Writes student data to a file.
- double rikiavimas_pagal_varda_laikas (vector< Studentas > studentai)

  Measures the time taken to sort a vector of students by first name in ascending order.
- double rikiavimas_pagal_pavarde_laikas (vector< Studentas > studentai)

  Measures the time taken to sort a vector of students by last name in ascending order.
- double rikiavimas_pagal_pazymius_laikas (vector< Studentas > studentai)

  Measures the time taken to sort a vector of students by their grades in ascending order.
- vector< Studentas > vargsiuku_atrinkimas (vector< Studentas > &studentai)

  Selects students whose final grade is less than 5 and returns them in a new vector.
- vector< Studentas > galvociu_atrinkimas (vector< Studentas > &studentai)

  Selects students whose final grade is greater than or equal to 5 and returns them in a new vector.
- void studentu_isskirstymas (string failo_pavadinimas)

  Sorts students by a selected criterion and splits them into two groups based on their final grades.
- void laiko_skaiciavimas_failo_generavimas (int studentu_skaicius, int nd_skaicius)

  Measures the time required to generate the student data file.
- void laiko_skaiciavimas (string failo_pavadinimas, int rikiavimo_pasirinkimas)

  Measures the total time taken to read, sort, and separate student data into different files.

5.1.1 Detailed Description

This file contains the implementation of a program for managing and processing student data.

5.1.2 Function Documentation

5.1.2.1 galvociu_atrinkimas()

```
vector< Studentas > galvociu_atrinkimas (
            vector< Studentas > & studentai)
```

Selects students whose final grade is greater than or equal to 5 and returns them in a new vector.

Filters students with a final grade of 5 or higher into a separate vector.

This function filters out the students whose final grade is greater than or equal to 5 and returns them in a new vector.

Parameters

studentai	A vector of student objects to filter.

Returns

A vector of students with final grades greater than or equal to 5.

5.1.2.2 irasymas_i_faila()

```
void irasymas_i_faila (
            vector< Studentas > & studentai,
            string failo_pav)
```

Writes student data to a file.

This function writes the data to a specified file.

Parameters

studentai	A vector of student objects to write to the file.
failo_pav	The name of the file to save the student data.

5.1.2.3 laiko_skaiciavimas()

```
void laiko_skaiciavimas (
            string failo_pavadinimas,
            int rikiavimo_pasirinkimas)
```

Measures the total time taken to read, sort, and separate student data into different files.

Calculates and prints the time required for file reading, sorting, and filtering operations.

This function calculates the time taken to read the student data from a file, sort the students by the selected criterion, separate them into two groups based on their final grades, and then write the groups to separate files. It provides a detailed breakdown of each step.

Parameters

failo_pavadinimas	The name of the file to read the students from.
rikiavimo_pasirinkimas	The sorting criterion (1: by name, 2: by last name, 3: by grade).

5.1.2.4 laiko_skaiciavimas_failo_generavimas()

```
void laiko_skaiciavimas_failo_generavimas (
            int studentu_skaicius,
            int nd_skaicius)
```

Measures the time required to generate the student data file.

Measures the time required for file generation.

This function calculates the average time taken to generate the student data file over multiple iterations.

Parameters

studentu_skaicius	The number of students to generate.
nd_skaicius	The number of assignments (ND) each student has.

5.1.2.5 pagrindinio_failo_generavimas()

```
void pagrindinio_failo_generavimas (
            int studentu_skaicius,
            int nd_skaicius)
```

Generates the main file with student data including names, grades, and exam scores.

Generates the main student data file.

This function generates a file that contains a list of students with their respective names, grades for multiple assignments (ND), and final exam grades (EGZ). The file is saved with a name based on the number of students.

Parameters

studentu_skaicius	The number of students to generate.
nd_skaicius	The number of assignments (ND) each student has.

5.1.2.6 rikiavimas_pagal_pavarde_laikas()

```
double rikiavimas_pagal_pavarde_laikas (
            vector< Studentas > studentai)
```

Measures the time taken to sort a vector of students by last name in ascending order.

Measures the time required to sort the student data by surname.

This function calculates the average time taken to sort the students vector by their last names in ascending order over multiple iterations.

Parameters

studentai	A vector of student objects to sort.

Returns

The average time in seconds taken to sort the vector.

5.1.2.7 rikiavimas_pagal_pazymius_laikas()

```
double rikiavimas_pagal_pazymius_laikas (
            vector< Studentas > studentai)
```

Measures the time taken to sort a vector of students by their grades in ascending order.

Measures the time required to sort the student data by grades.

This function calculates the average time taken to sort the students vector by their grades in ascending order over multiple iterations.

Parameters

studentai	A vector of student objects to sort.

Returns

The average time in seconds taken to sort the vector.

5.1.2.8 rikiavimas_pagal_varda_laikas()

```
double rikiavimas_pagal_varda_laikas (
            vector< Studentas > studentai)
```

Measures the time taken to sort a vector of students by first name in ascending order.

Measures the time required to sort the student data by name.

This function calculates the average time taken to sort the students vector by their first names in ascending order over multiple iterations.

Parameters

studentai	A vector of student objects to sort.

Returns

The average time in seconds taken to sort the vector.

5.1.2.9 studentu_isskirstymas()

```
void studentu_isskirstymas (
            string failo_pavadinimas)
```

Sorts students by a selected criterion and splits them into two groups based on their final grades.

Sorts students and divides them into two separate files based on their final grades.

This function allows the user to select a sorting criterion (name, last name, or grade), sorts the students accordingly, then separates them into two groups based on their final grades. The two groups are saved to separate files: "vargsiukai" for students with final grades below 5, and "galvociai" for students with final grades 5 or higher.

Parameters

failo_pavadinimas	The name of the file to read the students from.

5.1.2.10 vargsiuku_atrinkimas()

```
vector< Studentas > vargsiuku_atrinkimas (
            vector< Studentas > & studentai)
```

Selects students whose final grade is less than 5 and returns them in a new vector.

Filters students with a final grade less than 5 into a separate vector.

This function filters out the students whose final grade is below 5 and returns them in a new vector.

Parameters

studentai	A vector of student objects to filter.

Returns

A vector of students with final grades less than 5.

5.2 Failas.h File Reference

Header file containing function declarations for managing student data and performing file operations.

```
#include "Header.h"
```

Functions

- vector< Studentas > vargsiuku_atrinkimas (vector< Studentas > &studentai)
  Filters students with a final grade less than 5 into a separate vector.
- vector< Studentas > galvociu_atrinkimas (vector< Studentas > &studentai)
  Filters students with a final grade of 5 or higher into a separate vector.
- void laiko_skaiciavimas (string failo_pavadinimas, int rikiavimo_pasirinkimas)
  Calculates and prints the time required for file reading, sorting, and filtering operations.
- void irasymas_i_faila (vector< Studentas > &studentai, string failo_pav)
  Writes student data to a file.
- double rikiavimas_pagal_varda_laikas (vector< Studentas > studentai)
  Measures the time required to sort the student data by name.
- double rikiavimas_pagal_pavarde_laikas (vector< Studentas > studentai)
  Measures the time required to sort the student data by surname.
- void laiko_skaiciavimas_failo_generavimas (int studentu_skaicius, int nd_skaicius)
  Measures the time required for file generation.
- void pagrindinio_failo_generavimas (int studentu_skaicius, int nd_skaicius)
  Generates the main student data file.
- void studentu_isskirstymas (string failo_pavadinimas)
  Sorts students and divides them into two separate files based on their final grades.
- double rikiavimas_pagal_pazymius_laikas (vector< Studentas > studentai)
  Measures the time required to sort the student data by grades.

5.2.1 Detailed Description

Header file containing function declarations for managing student data and performing file operations.

5.2.2 Function Documentation

5.2.2.1 galvociu_atrinkimas()

```
vector< Studentas > galvociu_atrinkimas (
            vector< Studentas > & studentai)
```

Filters students with a final grade of 5 or higher into a separate vector.

Parameters

studentai	A vector of students to filter.

Returns

A vector of students with a final grade of 5 or higher.

Filters students with a final grade of 5 or higher into a separate vector.

This function filters out the students whose final grade is greater than or equal to 5 and returns them in a new vector.

Parameters

studentai	A vector of student objects to filter.

Returns

A vector of students with final grades greater than or equal to 5.

5.2.2.2 irasymas_i_faila()

```
void irasymas_i_faila (
            vector< Studentas > & studentai,
            string failo_pav)
```

Writes student data to a file.

Parameters

studentai	A vector of students to be written to the file.
failo_pav	The base name of the file to be written.

This function writes the data to a specified file.

Parameters

studentai	A vector of student objects to write to the file.
failo_pav	The name of the file to save the student data.

5.2.2.3 laiko_skaiciavimas()

```
void laiko_skaiciavimas (
            string failo_pavadinimas,
            int rikiavimo_pasirinkimas)
```

Calculates and prints the time required for file reading, sorting, and filtering operations.

Parameters

failo_pavadinimas	The name of the file containing student data.
rikiavimo_pasirinkimas	The criterion for sorting (1: name, 2: surname, 3: grade).

Calculates and prints the time required for file reading, sorting, and filtering operations.

This function calculates the time taken to read the student data from a file, sort the students by the selected criterion, separate them into two groups based on their final grades, and then write the groups to separate files. It provides a detailed breakdown of each step.

Parameters

failo_pavadinimas	The name of the file to read the students from.
rikiavimo_pasirinkimas	The sorting criterion (1: by name, 2: by last name, 3: by grade).

5.2.2.4 laiko_skaiciavimas_failo_generavimas()

```
void laiko_skaiciavimas_failo_generavimas (
            int studentu_skaicius,
            int nd_skaicius)
```

Measures the time required for file generation.

Parameters

studentu_skaicius	The number of students.
nd_skaicius	The number of assignments.

Measures the time required for file generation.

This function calculates the average time taken to generate the student data file over multiple iterations.

Parameters

studentu_skaicius	The number of students to generate.
nd_skaicius	The number of assignments (ND) each student has.

5.2.2.5 pagrindinio_failo_generavimas()

```
void pagrindinio_failo_generavimas (
            int studentu_skaicius,
            int nd_skaicius)
```

Generates the main student data file.

Parameters

studentu_skaicius	The number of students to include in the file.
nd_skaicius	The number of assignments for each student.

Generates the main student data file.

This function generates a file that contains a list of students with their respective names, grades for multiple assignments (ND), and final exam grades (EGZ). The file is saved with a name based on the number of students.

Parameters

studentu_skaicius	The number of students to generate.
nd_skaicius	The number of assignments (ND) each student has.

5.2.2.6 rikiavimas_pagal_pavarde_laikas()

```
double rikiavimas_pagal_pavarde_laikas (
            vector< Studentas > studentai)
```

Measures the time required to sort the student data by surname.

Parameters

studentai	A vector of students to be sorted.

Returns

The time taken to perform the sorting, in seconds.

Measures the time required to sort the student data by surname.

This function calculates the average time taken to sort the students vector by their last names in ascending order over multiple iterations.

Parameters

studentai	A vector of student objects to sort.

Returns

The average time in seconds taken to sort the vector.

5.2.2.7 rikiavimas_pagal_pazymius_laikas()

```
double rikiavimas_pagal_pazymius_laikas (
            vector< Studentas > studentai)
```

Measures the time required to sort the student data by grades.

Parameters

studentai	A vector of students to be sorted.

Returns

The time taken to perform the sorting, in seconds.

Measures the time required to sort the student data by grades.

This function calculates the average time taken to sort the students vector by their grades in ascending order over multiple iterations.

Parameters

studentai	A vector of student objects to sort.

Returns

The average time in seconds taken to sort the vector.

5.2.2.8 rikiavimas_pagal_varda_laikas()

```
double rikiavimas_pagal_varda_laikas (
            vector< Studentas > studentai)
```

Measures the time required to sort the student data by name.

Parameters

studentai	A vector of students to be sorted.

Returns

The time taken to perform the sorting, in seconds.

Measures the time required to sort the student data by name.

This function calculates the average time taken to sort the students vector by their first names in ascending order over multiple iterations.

Parameters

studentai	A vector of student objects to sort.

Returns

The average time in seconds taken to sort the vector.

5.2.2.9 studentu_isskirstymas()

```
void studentu_isskirstymas (
            string failo_pavadinimas)
```

Sorts students and divides them into two separate files based on their final grades.

Parameters

failo_pavadinimas	The name of the file containing the student data to be processed.

Sorts students and divides them into two separate files based on their final grades.

This function allows the user to select a sorting criterion (name, last name, or grade), sorts the students accordingly, then separates them into two groups based on their final grades. The two groups are saved to separate files: "vargsiukai" for students with final grades below 5, and "galvociai" for students with final grades 5 or higher.

Parameters

failo_pavadinimas	The name of the file to read the students from.

5.2.2.10 vargsiuku_atrinkimas()

```
vector< Studentas > vargsiuku_atrinkimas (
            vector< Studentas > & studentai)
```

Filters students with a final grade less than 5 into a separate vector.

Parameters

studentai	A vector of students to filter.

Returns

A vector of students with a final grade less than 5.

Filters students with a final grade less than 5 into a separate vector.

This function filters out the students whose final grade is below 5 and returns them in a new vector.

Parameters

studentai	A vector of student objects to filter.

Returns

A vector of students with final grades less than 5.

5.3 Failas.h

Go to the documentation of this file.

```
00001
00005 #ifndef Failas_h
00006 #define Failas_h
00007 #include "Header.h"
00008
00014 vector<Studentas> vargsiuku_atrinkimas(vector<Studentas>& studentai);
00015
00021 vector<Studentas> galvociu_atrinkimas(vector<Studentas>& studentai);
00022
00028 void laiko_skaiciavimas(string failo_pavadinimas, int rikiavimo_pasirinkimas);
00029
00035 void irasymas_i_faila(vector<Studentas>& studentai, string failo_pav);
00036
00042 double rikiavimas_pagal_varda_laikas(vector<Studentas> studentai);
00043
00049 double rikiavimas_pagal_pavarde_laikas(vector<Studentas> studentai);
00050
00056 void laiko_skaiciavimas_failo_generavimas(int studentu_skaicius, int nd_skaicius);
00057
00063 void pagrindinio_failo_generavimas(int studentu_skaicius, int nd_skaicius);
00064
00069 void studentu_isskirstymas(string failo_pavadinimas);
00070
00076 double rikiavimas_pagal_pazymius_laikas(vector<Studentas> studentai);
00077
00078
00079 #endif /* Failas_h */
```

5.4 Header.h File Reference

```
#include <iostream>
#include <vector>
#include <numeric>
#include <algorithm>
#include <iomanip>
#include <random>
#include <fstream>
#include <sstream>
#include <string>
#include <cstdlib>
#include <thread>
#include <cstdio>
#include <list>
```

5.5 Header.h

Go to the documentation of this file.

```
00001 #ifndef Header_h
00002 #define Header_h
00003
00004
00005 #include <iostream>
00006 #include <vector> // vector<> sukurimui
00007 #include <numeric> // accumulate() funkcijos naudojimui
00008 #include <algorithm> // sort() funkcijos naudojimui
00009 #include <iomanip> // setw() funkcijos naudojimui
00010 #include <random> // random_device(), mt19937(), uniform_int_distribution() funkciju naudojimui
00011 #include <fstream> // Skaitymui is failo
00012 #include <sstream> // stringstream() funkcijos naudojimui
00013 #include <string> // string manipuliavimui
00014 #include <cstdlib> // stoi() funkcijos naudojimui
00015 #include <thread>
00016 #include <cstdio>
00017 #include <list>
00018
00019
00020 using std::cout;
00021 using std::cin;
00022 using std::endl;
00023 using std::string;
00024 using std::vector;
00025 using std::random_device;
00026 using std::mt19937;
00027 using std::uniform_int_distribution;
00028 using std::setw;
00029 using std::left;
00030 using std::ifstream;
00031 using std::stringstream;
00032 using std::ofstream;
00033 using std::runtime_error;
00034 using std::exception;
00035 using std::cerr;
00036 using std::invalid_argument;
00037 using std::out_of_range;
00038 using std::fixed;
00039 using std::setprecision;
00040 using std::to_string;
00041 using std::chrono::high_resolution_clock;
00042 using std::chrono::duration_cast;
00043 using std::chrono::duration;
00044 using std::chrono::milliseconds;
00045 using std::list;
00046
00047
00048
00049 #endif /* Header_h */
```

5.6 main.cpp File Reference

```
#include "Header.h"
#include "Studentas.h"
#include "Failas.h"
#include "v0_3_header.h"
#include "strategijos.h"
```

Functions

- int main (int argc, const char *argv[])

  The entry point of the program.

5.6.1 Function Documentation

5.6.1.1 main()

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

The entry point of the program.

This main function demonstrates various operations, including reading data from files, generating data, partitioning students based on their grades, and measuring execution time for different strategies and container types.

5.7 strategijos.cpp File Reference

This file contains functions for processing student data with different strategies for reading, sorting, filtering, and writing to files.

```
#include <stdio.h>
#include "v0_3_header.h"
#include "Studentas.h"
#include "Header.h"
#include "Failas.h"
#include "strategijos.h"
```

Functions

- bool operator== (const Studentas &a, const Studentas &b)

  Compares two Studentas objects for equality.
- void skaidymas_2_strategija_vector (string failo_pavadinimas, int rikiavimo_pasirinkimas)

  Splits students into two categories using strategy 2 with vector container.
- void skaidymas_2_strategija_list (string failo_pavadinimas, int rikiavimo_pasirinkimas)

  Splits students into two categories using strategy 2 with list container.
- void skaidymas_3_strategija_vector (string failo_pavadinimas, int rikiavimo_pasirinkimas)

  Splits students into two categories using strategy 3 with vector container.
- void skaidymas_3_strategija_list (string failo_pavadinimas, int rikiavimo_pasirinkimas)

  Splits students into two categories using strategy 3 with list container.

5.7.1 Detailed Description

This file contains functions for processing student data with different strategies for reading, sorting, filtering, and writing to files.

5.7.2 Function Documentation

5.7.2.1 operator==()

```
bool operator== (
            const Studentas & a,
            const Studentas & b)
```

Compares two Studentas objects for equality.

Compares two `Studentas` objects for equality.

This function is used to compare two Studentas objects to determine if they are equal based on their names and surnames. It is mainly used in the context of erasing students in the container.

Parameters

a	The first Studentas object.
b	The second Studentas object.

Returns

true if both students have the same name and surname, otherwise false.

5.7.2.2 skaidymas_2_strategija_list()

```
void skaidymas_2_strategija_list (
            string failo_pavadinimas,
            int rikiavimo_pasirinkimas)
```

Splits students into two categories using strategy 2 with list container.

Function for processing student data with a list using the second strategy.

This function is similar to `skaidymas_2_strategija_vector` but works with the list container. It reads students from a file, splits them into two groups (vargsiukai and others), sorts the groups based on a selected criterion (name, surname, or grades), and writes them to separate files. The function also measures and reports the time taken for reading, splitting, sorting, and writing the data.

Parameters

failo_pavadinimas	The name of the input file.
rikiavimo_pasirinkimas	The sorting criterion (1 for name, 2 for surname, 3 for grades).

5.7.2.3 skaidymas_2_strategija_vector()

```
void skaidymas_2_strategija_vector (
            string failo_pavadinimas,
            int rikiavimo_pasirinkimas)
```

Splits students into two categories using strategy 2 with vector container.

Function for processing student data with a vector using the second strategy.

This function reads students from a file, divides them into two groups (vargsiukai and others), sorts them based on a chosen criterion (name, surname, or grade), and writes them back to separate files. It also measures and reports the time taken for reading, splitting, sorting, and writing the data.

Parameters

failo_pavadinimas	The name of the input file.
rikiavimo_pasirinkimas	The sorting criterion (1 for name, 2 for surname, 3 for grades).

5.7.2.4 skaidymas_3_strategija_list()

```
void skaidymas_3_strategija_list (
            string failo_pavadinimas,
            int rikiavimo_pasirinkimas)
```

Splits students into two categories using strategy 3 with list container.

Function for processing student data with a list using the second strategy.

Parameters

failo_pavadinimas	The name of the input file.
rikiavimo_pasirinkimas	The sorting criterion (1 for name, 2 for surname, 3 for grades).

5.7.2.5 skaidymas_3_strategija_vector()

```
void skaidymas_3_strategija_vector (
            string failo_pavadinimas,
            int rikiavimo_pasirinkimas)
```

Splits students into two categories using strategy 3 with vector container.

Function for processing student data with a vector using the third strategy.

This function is based on the principles of strategy 1 but works with the vector container. It reads students from a file, compares and separetes them, then sorts the them based on a selected criterion (name, surname, or grades), and writes the groups to separate files. The function also measures and reports the time taken for reading, splitting, sorting, and writing the data.

Parameters

failo_pavadinimas	The name of the input file.
rikiavimo_pasirinkimas	The sorting criterion (1 for name, 2 for surname, 3 for grades).

5.8 strategijos.h File Reference

This header file defines the function prototypes for various strategies related to processing student data using different containers (vector or list).

```
#include "Header.h"
#include <stdio.h>
#include "Studentas.h"
#include "Failas.h"
#include "v0_3_header.h"
```

Functions

- void skaidymas_2_strategija_vector (string failo_pavadinimas, int rikiavimo_pasirinkimas)

  Function for processing student data with a vector using the second strategy.
- void skaidymas_2_strategija_list (string failo_pavadinimas, int rikiavimo_pasirinkimas)

  Function for processing student data with a list using the second strategy.
- void skaidymas_3_strategija_vector (string failo_pavadinimas, int rikiavimo_pasirinkimas)

  Function for processing student data with a vector using the third strategy.
- void skaidymas_3_strategija_list (string failo_pavadinimas, int rikiavimo_pasirinkimas)

  Function for processing student data with a list using the second strategy.
- bool operator== (const Studentas &a, const Studentas &b)

  Compares two `Studentas` objects for equality.

5.8.1 Detailed Description

This header file defines the function prototypes for various strategies related to processing student data using different containers (vector or list).

5.8.2 Function Documentation

5.8.2.1 operator==()

```
bool operator== (
            const Studentas & a,
            const Studentas & b)
```

Compares two `Studentas` objects for equality.

This operator is used to compare two `Studentas` objects, likely based on student attributes such as name, surname, or grade.

Parameters

a	The first student to compare.
b	The second student to compare.

Returns

Returns `true` if both students are considered equal, otherwise `false`.

Compares two `Studentas` objects for equality.

This function is used to compare two Studentas objects to determine if they are equal based on their names and surnames. It is mainly used in the context of erasing students in the container.

Parameters

a	The first Studentas object.
b	The second Studentas object.

Returns

true if both students have the same name and surname, otherwise false.

5.8.2.2 skaidymas_2_strategija_list()

```
void skaidymas_2_strategija_list (
            string failo_pavadinimas,
            int rikiavimo_pasirinkimas)
```

Function for processing student data with a list using the second strategy.

Parameters

failo_pavadinimas	The name of the input file containing student data.
rikiavimo_pasirinkimas	An integer representing the sorting criteria.

Function for processing student data with a list using the second strategy.

This function is similar to `skaidymas_2_strategija_vector` but works with the list container. It reads students from a file, splits them into two groups (vargsiukai and others), sorts the groups based on a selected criterion (name, surname, or grades), and writes them to separate files. The function also measures and reports the time taken for reading, splitting, sorting, and writing the data.

Parameters

failo_pavadinimas	The name of the input file.
rikiavimo_pasirinkimas	The sorting criterion (1 for name, 2 for surname, 3 for grades).

5.8.2.3 skaidymas_2_strategija_vector()

```
void skaidymas_2_strategija_vector (
             string failo_pavadinimas,
             int rikiavimo_pasirinkimas)
```

Function for processing student data with a vector using the second strategy.

Parameters

failo_pavadinimas	The name of the input file containing student data.
rikiavimo_pasirinkimas	An integer representing the sorting criteria.

Function for processing student data with a vector using the second strategy.

This function reads students from a file, divides them into two groups (vargsiukai and others), sorts them based on a chosen criterion (name, surname, or grade), and writes them back to separate files. It also measures and reports the time taken for reading, splitting, sorting, and writing the data.

Parameters

failo_pavadinimas	The name of the input file.
rikiavimo_pasirinkimas	The sorting criterion (1 for name, 2 for surname, 3 for grades).

5.8.2.4 skaidymas_3_strategija_list()

```
void skaidymas_3_strategija_list (
             string failo_pavadinimas,
             int rikiavimo_pasirinkimas)
```

Function for processing student data with a list using the second strategy.

Parameters

failo_pavadinimas	The name of the input file containing student data.
rikiavimo_pasirinkimas	An integer representing the sorting criteria.

Function for processing student data with a list using the second strategy.

Parameters

failo_pavadinimas	The name of the input file.
rikiavimo_pasirinkimas	The sorting criterion (1 for name, 2 for surname, 3 for grades).

5.8.2.5 skaidymas_3_strategija_vector()

```
void skaidymas_3_strategija_vector (
            string failo_pavadinimas,
            int rikiavimo_pasirinkimas)
```

Function for processing student data with a vector using the third strategy.

Parameters

failo_pavadinimas	The name of the input file containing student data.
rikiavimo_pasirinkimas	An integer representing the sorting criteria.

Function for processing student data with a vector using the third strategy.

This function is based on the principles of strategy 1 but works with the vector container. It reads students from a file, compares and separetes them, then sorts the them based on a selected criterion (name, surname, or grades), and writes the groups to separate files. The function also measures and reports the time taken for reading, splitting, sorting, and writing the data.

Parameters

failo_pavadinimas	The name of the input file.
rikiavimo_pasirinkimas	The sorting criterion (1 for name, 2 for surname, 3 for grades).

5.9 strategijos.h

Go to the documentation of this file.

```
00001
00006 #ifndef strategijos_h
00007 #define strategijos_h
00008 #include "Header.h"
00009 #include <stdio.h>
00010 #include "Studentas.h"
00011 #include "Failas.h"
00012 #include "v0_3_header.h"
00013
00014
00020 void skaidymas_2_strategija_vector(string failo_pavadinimas, int rikiavimo_pasirinkimas);
00021
```

```
00027 void skaidymas_2_strategija_list(string failo_pavadinimas, int rikiavimo_pasirinkimas);
00028
00034 void skaidymas_3_strategija_vector(string failo_pavadinimas, int rikiavimo_pasirinkimas);
00035
00041 void skaidymas_3_strategija_list(string failo_pavadinimas, int rikiavimo_pasirinkimas);
00042
00053 bool operator==(const Studentas& a, const Studentas& b);
00054
00055
00056
00057
00058
00059
00060 #endif /* strategijos_h */
```

5.10 Studentas.cpp File Reference

Implementation of the Studentas class and related functions.

```
#include <stdio.h>
#include "Studentas.h"
#include "Header.h"
#include "v0_3_header.h"
#include "Zmogus.h"
```

Functions

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

  Overloaded output stream operator for Studentas.
- std::istream & operator>> (std::istream &is, Studentas &s)

  Overloaded input stream operator for Studentas.
- void rule_of_three_metodu_demonstracija ()

  Demonstrates the use of rule-of-three methods (constructor, copy, and assignment).
- void ivesties_isvesties_metodu_demonstracija ()

  Demonstrates input and output operators for the Studentas class.
- vector< int > random_pazymiai (int pazymiu_sk)

  Generates a vector of random grades.
- int random_egz ()

  Generates a random exam grade.
- void ivedimas (Studentas &Lok)

  Function to input new students;.
- void valymas (Studentas &Lok)

  Function to clear certain elements of a Studentas object. Ensures no residual data is printed alongside.
- bool palyginti_pavardes (const Studentas &a, const Studentas &b)

  Comparator function to compare students by last names.
- bool palyginti_vardus (const Studentas &a, const Studentas &b)

  Comparator function to compare students by first names.
- bool palyginti_pazymius (const Studentas &a, const Studentas &b)

  Comparator function to compare students by their final grade.
- void isvedimas (vector< Studentas > studentai)

  Function to print student data (final grade calculated with average).
- void isvedimas_su_vidurkiu (vector< Studentas > studentai)

  Function to print student data (final grade calculated with average).
- void isvedimas_su_mediana (vector< Studentas > studentai)

Function to print student data (final grade calculated with median).

- vector< Studentas > skaitymas_is_failo (vector< Studentas > studentai, string failo_pav)

  Reads students' data from a file.
- void irasymas (vector< Studentas > &studentai, string failo_pav, int pasirinkimas)

  Sorts and writes student data to a file.
- void irasymas_list (list< Studentas > &studentai, string failo_pav, int pasirinkimas)

  Sorts and writes student data to a file.
- void isvedimas_list (list< Studentas > studentai)

  Function to print student data (final grade calculated with average).

5.10.1 Detailed Description

Implementation of the Studentas class and related functions.

5.10.2 Function Documentation

5.10.2.1 irasymas()

```
void irasymas (
            vector< Studentas > & studentai,
            string failo_pav,
            int pasirinkimas)
```

Sorts and writes student data to a file.

Writes student data to a file.

Parameters

studentai	Vector of Studentas objects.
failo_pav	File name to write data to.
pasirinkimas	Sorting option.

5.10.2.2 irasymas_list()

```
void irasymas_list (
            list< Studentas > & studentai,
            string failo_pav,
            int pasirinkimas)
```

Sorts and writes student data to a file.

Writes student data to a file using a list.

Parameters

studentai	Vector of Studentas objects.
failo_pav	File name to write data to.
pasirinkimas	Sorting option.

5.10.2.3 isvedimas()

```
void isvedimas (
            vector< Studentas > studentai)
```

Function to print student data (final grade calculated with average).

Outputs a list of students.

Parameters

studentai	Vector of Studentas objects.

5.10.2.4 isvedimas_list()

```
void isvedimas_list (
            list< Studentas > studentai)
```

Function to print student data (final grade calculated with average).

Outputs a list of students.

Parameters

studentai	Vector of Studentas objects.

5.10.2.5 isvedimas_su_mediana()

```
void isvedimas_su_mediana (
            vector< Studentas > studentai)
```

Function to print student data (final grade calculated with median).

Outputs students with their final grade based on median.

Parameters

studentai	Vector of Studentas objects.

5.10.2.6 isvedimas_su_vidurkiu()

```
void isvedimas_su_vidurkiu (
            vector< Studentas > studentai)
```

Function to print student data (final grade calculated with average).

Outputs students with their final grade based on average.

Parameters

studentai	Vector of Studentas objects.

5.10.2.7 ivedimas()

```
void ivedimas (
            Studentas & Lok)
```

Function to input new students;.

Inputs student information.

Parameters

Lok	Reference to a Studentas object.

5.10.2.8 ivesties_isvesties_metodu_demonstracija()

```
void ivesties_isvesties_metodu_demonstracija ()
```

Demonstrates input and output operators for the Studentas class.

Demonstrates input and output methods.

5.10.2.9 operator<<()

```
std::ostream & operator<< (
            std::ostream & os,
            const Studentas & s)
```

Overloaded output stream operator for Studentas.

Output stream operator for Studentas.

Parameters

os	The output stream.
s	The Studentas object to output.

Returns

A reference to the output stream.

5.10.2.10 operator>>()

```
std::istream & operator>> (
            std::istream & is,
            Studentas & s)
```

Overloaded input stream operator for Studentas.

Input stream operator for Studentas.

Parameters

is	The input stream.
s	The Studentas object to populate.

Returns

A reference to the input stream.

5.10.2.11 palyginti_pavardes()

```
bool palyginti_pavardes (
            const Studentas & a,
            const Studentas & b)
```

Comparator function to compare students by last names.

Compares two students by last name.

Parameters

a	First Studentas object to compare.
b	Second Studentas object to compare.

Returns

True if the last name of a comes before b lexicographically.

5.10.2.12 palyginti_pazymius()

```
bool palyginti_pazymius (
            const Studentas & a,
            const Studentas & b)
```

Comparator function to compare students by their final grade.

Compares two students by their grades.

Parameters

a	First Studentas object to compare.
b	Second Studentas object to compare.

Returns

True if the final average grade of a is less than b.

5.10.2.13 palyginti_vardus()

```
bool palyginti_vardus (
            const Studentas & a,
            const Studentas & b)
```

Comparator function to compare students by first names.

Compares two students by first name.

Parameters

a	First Studentas object to compare.
b	Second Studentas object to compare.

Returns

True if the first name of a comes before b lexicographically.

5.10.2.14 random_egz()

```
int random_egz ()
```

Generates a random exam grade.

Returns

A random grade between 1 and 10.

5.10.2.15 random_pazymiai()

```
vector< int > random_pazymiai (
            int pazymiu_sk)
```

Generates a vector of random grades.

Generates random grades for a student.

Parameters

pazymiu_sk	Number of grades to generate.

Returns

A vector containing random grades.

5.10.2.16 rule_of_three_metodu_demonstracija()

```
void rule_of_three_metodu_demonstracija ()
```

Demonstrates the use of rule-of-three methods (constructor, copy, and assignment).

Demonstrates the rule of three methods.

5.10.2.17 skaitymas_is_failo()

```
vector< Studentas > skaitymas_is_failo (
            vector< Studentas > studentai,
            string failo_pav)
```

Reads students' data from a file.

Reads students from a file.

Parameters

studentai	Vector of Studentas objects.
failo_pav	File name to read data from.

Returns

Updated vector of Studentas objects.

5.10.2.18 valymas()

```
void valymas (
            Studentas & Lok)
```

Function to clear certain elements of a Studentas object. Ensures no residual data is printed alongside.

Cleans up the student data.

Parameters

Lok	Reference to a Studentas object.

5.11 Studentas.h File Reference

Declaration of the `Studentas` class.

```
#include "Header.h"
#include "Zmogus.h"
```

Classes

- class Studentas

  Represents a student, inheriting from the Zmogus class.

Functions

- vector< int > random_pazymiai (int pazymiu_sk)

  Generates random grades for a student.
- void ivedimas (Studentas &Lok)

  Inputs student information.
- void valymas (Studentas &Lok)

  Cleans up the student data.
- bool palyginti_pavardes (const Studentas &a, const Studentas &b)

  Compares two students by last name.
- bool palyginti_vardus (const Studentas &a, const Studentas &b)

  Compares two students by first name.

- void isvedimas (vector< Studentas > studentai)

  Outputs a list of students.
- void isvedimas_su_vidurkiu (vector< Studentas > studentai)

  Outputs students with their final grade based on average.
- void isvedimas_su_mediana (vector< Studentas > studentai)

  Outputs students with their final grade based on median.
- vector< Studentas > skaitymas_is_failo (vector< Studentas > studentai, string failo_pav)

  Reads students from a file.
- void irasymas (vector< Studentas > &studentai, string failo_pav, int pasirinkimas)

  Writes student data to a file.
- int random_egz ()

  Generates a random exam grade.
- void irasymas_list (list< Studentas > &studentai, string failo_pav, int pasirinkimas)

  Writes student data to a file using a list.
- void isvedimas_list (list< Studentas > studentai)

  Outputs a list of students.
- bool palyginti_pazymius (const Studentas &a, const Studentas &b)

  Compares two students by their grades.
- void rule_of_three_metodu_demonstracija ()

  Demonstrates the rule of three methods.
- void ivesties_isvesties_metodu_demonstracija ()

  Demonstrates input and output methods.

5.11.1 Detailed Description

Declaration of the `Studentas` class.

5.11.2 Function Documentation

5.11.2.1 irasymas()

```
void irasymas (
            vector< Studentas > & studentai,
            string failo_pav,
            int pasirinkimas)
```

Writes student data to a file.

Parameters

studentai	Vector of students to write.
failo_pav	File name to write to.
pasirinkimas	Option for file writing.

Writes student data to a file.

Parameters

studentai	Vector of Studentas objects.
failo_pav	File name to write data to.
pasirinkimas	Sorting option.

5.11.2.2 irasymas_list()

```
void irasymas_list (
            list< Studentas > & studentai,
            string failo_pav,
            int pasirinkimas)
```

Writes student data to a file using a list.

Parameters

studentai	List of students to write.
failo_pav	File name to write to.
pasirinkimas	Option for file writing.

Writes student data to a file using a list.

Parameters

studentai	Vector of Studentas objects.
failo_pav	File name to write data to.
pasirinkimas	Sorting option.

5.11.2.3 isvedimas()

```
void isvedimas (
            vector< Studentas > studentai)
```

Outputs a list of students.

Parameters

studentai	Vector of students to output.

Outputs a list of students.

Parameters

studentai	Vector of Studentas objects.

5.11.2.4 isvedimas_list()

```
void isvedimas_list (
            list< Studentas > studentai)
```

Outputs a list of students.

Parameters

studentai	List of students to output.

Outputs a list of students.

Parameters

studentai	Vector of Studentas objects.

5.11.2.5 isvedimas_su_mediana()

```
void isvedimas_su_mediana (
            vector< Studentas > studentai)
```

Outputs students with their final grade based on median.

Parameters

studentai	Vector of students to output.

Outputs students with their final grade based on median.

Parameters

studentai	Vector of Studentas objects.

5.11.2.6 isvedimas_su_vidurkiu()

```
void isvedimas_su_vidurkiu (
            vector< Studentas > studentai)
```

Outputs students with their final grade based on average.

Parameters

studentai	Vector of students to output.

Outputs students with their final grade based on average.

Parameters

studentai	Vector of Studentas objects.

5.11.2.7 ivedimas()

```
void ivedimas (
            Studentas & Lok)
```

Inputs student information.

Parameters

Lok	Studentas object to populate.

Inputs student information.

Parameters

Lok	Reference to a Studentas object.

5.11.2.8 ivesties_isvesties_metodu_demonstracija()

```
void ivesties_isvesties_metodu_demonstracija ()
```

Demonstrates input and output methods.

Demonstrates input and output methods.

5.11.2.9 palyginti_pavardes()

```
bool palyginti_pavardes (
            const Studentas & a,
            const Studentas & b)
```

Compares two students by last name.

Parameters

a	First student.
b	Second student.

Returns

True if a's last name is less than b's.

Compares two students by last name.

Parameters

a	First Studentas object to compare.
b	Second Studentas object to compare.

Returns

True if the last name of a comes before b lexicographically.

5.11.2.10 palyginti_pazymius()

```
bool palyginti_pazymius (
            const Studentas & a,
            const Studentas & b)
```

Compares two students by their grades.

Parameters

a	First student.
b	Second student.

Returns

True if a's grades are less than b's.

Compares two students by their grades.

Parameters

a	First Studentas object to compare.
b	Second Studentas object to compare.

Returns

True if the final average grade of a is less than b.

5.11.2.11 palyginti_vardus()

```
bool palyginti_vardus (
            const Studentas & a,
            const Studentas & b)
```

Compares two students by first name.

Parameters

a	First student.
b	Second student.

Returns

True if a's first name is less than b's.

Compares two students by first name.

Parameters

a	First Studentas object to compare.
b	Second Studentas object to compare.

Returns

True if the first name of a comes before b lexicographically.

5.11.2.12 random_egz()

```
int random_egz ()
```

Generates a random exam grade.

Returns

Random exam grade.

A random grade between 1 and 10.

5.11.2.13 random_pazymiai()

```
vector< int > random_pazymiai (
            int pazymiu_sk)
```

Generates random grades for a student.

Parameters

pazymiu_sk	Number of grades to generate.

Returns

Vector of random grades.

Generates random grades for a student.

Parameters

pazymiu_sk	Number of grades to generate.

Returns

A vector containing random grades.

5.11.2.14 rule_of_three_metodu_demonstracija()

```
void rule_of_three_metodu_demonstracija ()
```

Demonstrates the rule of three methods.

Demonstrates the rule of three methods.

5.11.2.15 skaitymas_is_failo()

```
vector< Studentas > skaitymas_is_failo (
            vector< Studentas > studentai,
            string failo_pav)
```

Reads students from a file.

Parameters

studentai	Vector to store read students.
failo_pav	File name to read from.

Returns

Vector of students.

Reads students from a file.

Parameters

studentai	Vector of Studentas objects.
failo_pav	File name to read data from.

Returns

Updated vector of Studentas objects.

5.11.2.16 valymas()

```
void valymas (
              Studentas & Lok)
```

Cleans up the student data.

Parameters

Lok	Studentas object to clear.

Cleans up the student data.

Parameters

Lok	Reference to a Studentas object.

5.12 Studentas.h

Go to the documentation of this file.

```
00001
00005 #ifndef Studentas_h
00006 #define Studentas_h
00007
00008 #include "Header.h"
00009 #include "Zmogus.h"
00010
00015 class Studentas : public Zmogus {
00016 private:
00020     vector<int> namu_darbai;
00021
```

```
00025     int egzaminas;
00026
00030     float pazymiu_vidurkis;
00031
00035     float mediana;
00036
00040     float galutinis_vid;
00041
00045     float galutinis_med;
00046
00047 public:
00051     Studentas();
00052
00060     Studentas(string vardas, string pavarde, vector<int> namu_darbai, int egzaminas);
00061
00066     Studentas(const Studentas& other);
00067
00073     Studentas& operator=(const Studentas& other);
00074
00078     ~Studentas();
00079
00084     void setVardas(const string& v);
00085
00090     void setPavarde(const std::string& p);
00091
00096     void setNamuDarbai(const std::vector<int>& nd);
00097
00102     void setEgzaminas(int egz);
00103
00108     const std::string& getVardas() const override;
00109
00114     const std::string& getPavarde() const override;
00115
00120     const std::vector<int>& getNamuDarbai() const;
00121
00126     int getEgzaminas() const;
00127
00132     double getGalutinisVid() const;
00133
00138     double getGalutinisMed() const;
00139
00143     void vidurkioSkaiciavimas();
00144
00148     void medianosSkaiciavimas();
00149
00156     friend std::ostream& operator«(std::ostream& os, const Studentas& s);
00157
00164     friend std::istream& operator»(std::istream& is, Studentas& s);
00165 };
00166
00172 vector<int> random_pazymiai(int pazymiu_sk);
00173
00178 void ivedimas(Studentas & Lok);
00179
00184 void valymas(Studentas & Lok);
00185
00192 bool palyginti_pavardes(const Studentas& a, const Studentas& b);
00193
00200 bool palyginti_vardus(const Studentas& a, const Studentas& b);
00201
00206 void isvedimas(vector<Studentas> studentai);
00207
00212 void isvedimas_su_vidurkiu(vector<Studentas> studentai);
00213
00218 void isvedimas_su_mediana(vector<Studentas> studentai);
00219
00226 vector<Studentas> skaitymas_is_failo(vector<Studentas> studentai, string failo_pav);
00227
00234 void irasymas(vector<Studentas>& studentai, string failo_pav, int pasirinkimas);
00235
00240 int random_egz();
00241
00248 void irasymas_list(list<Studentas>& studentai, string failo_pav, int pasirinkimas);
00249
00254 void isvedimas_list(list<Studentas> studentai);
00255
00262 bool palyginti_pazymius(const Studentas& a, const Studentas& b);
00263
00267 void rule_of_three_metodu_demonstracija();
00268
00272 void ivesties_isvesties_metodu_demonstracija();
00273
00274 #endif /* Studentas_h */
00275
```

5.13 v0_3.cpp File Reference

```
#include "v0_3_header.h"
#include "Studentas.h"
#include "Header.h"
#include "Failas.h"
```

Functions

- list< Studentas > skaitymas_is_failo_list (list< Studentas > &studentai, string failo_pav)
  Reads student data from a file into a list of Studentas objects.
- list< Studentas > vargsiuku_atrinkimas_naudojant_list (list< Studentas > &studentai)
  Filters students with a final grade below 5.
- list< Studentas > galvociu_atrinkimas_naudojant_list (list< Studentas > &studentai)
  Filters students with a final grade of 5 or higher.
- double rikiavimas_pagal_varda_laikas_list (list< Studentas > studentai)
  Measures the time to sort students by name in ascending order.
- double rikiavimas_pagal_pavarde_laikas_list (list< Studentas > studentai)
  Measures the time to sort students by surname in ascending order.
- double rikiavimas_pagal_pazymius_laikas_list (list< Studentas > studentai)
  Measures the time to sort students by grade in ascending order.
- void irasymas_naudojant_list (list< Studentas > &studentai, string failo_pav)
  Writes the filtered students (vargsiukai) and (galvociai) to separate files.
- void studentu_isskirstymas_list (string failo_pavadinimas)
  Manages the entire student data processing workflow.
- void laiko_skaiciavimas_list_konteineris (string failo_pavadinimas, int rikiavimo_pasirinkimas)
  Measures and outputs the execution time for the entire student data processing.

5.13.1 Function Documentation

5.13.1.1 galvociu_atrinkimas_naudojant_list()

```
list< Studentas > galvociu_atrinkimas_naudojant_list (
            list< Studentas > & studentai)
```

Filters students with a final grade of 5 or higher.

Filters students into a list of "galvociai" (students with high grades).

This function creates a list of students who have a final grade of 5 or higher.

Parameters

studentai	The list of Studentas objects to filter.

Returns

A new list of students with a final grade of 5 or higher.

5.13.1.2 irasymas_naudojant_list()

```
void irasymas_naudojant_list (
            list< Studentas > & studentai,
            string failo_pav)
```

Writes the filtered students (vargsiukai) and (galvociai) to separate files.

Writes student data to a file using a `std::list`.

This function saves the filtered lists of students (those with final grades below 5 and those with final grades of 5 or higher) to files.

Parameters

studentai	The list of Studentas objects to write.
failo_pav	The name of the file to write the data to.

5.13.1.3 laiko_skaiciavimas_list_konteineris()

```
void laiko_skaiciavimas_list_konteineris (
            string failo_pavadinimas,
            int rikiavimo_pasirinkimas)
```

Measures and outputs the execution time for the entire student data processing.

Measures execution times for list operations, including sorting and filtering.

This function calculates and displays the time taken to read the student data from a file, sort them, filter them, and write the results into separate files.

Parameters

failo_pavadinimas	The name of the input file.
rikiavimo_pasirinkimas	The sorting criterion (1 for name, 2 for surname, 3 for grade).

5.13.1.4 rikiavimas_pagal_pavarde_laikas_list()

```
double rikiavimas_pagal_pavarde_laikas_list (
            list< Studentas > studentai)
```

Measures the time to sort students by surname in ascending order.

Measures the time required to sort students by surname in ascending order.

This function measures the time it takes to sort the list of students by their surnames in ascending order. It performs the sort 5 times and returns the average time taken in seconds.

Parameters

studentai	The list of Studentas objects to sort.

Returns

The average time in seconds to sort the students by surname.

5.13.1.5 rikiavimas_pagal_pazymius_laikas_list()

```
double rikiavimas_pagal_pazymius_laikas_list (
            list< Studentas > studentai)
```

Measures the time to sort students by grade in ascending order.

Measures the time required to sort students by grades in ascending order.

This function measures the time it takes to sort the list of students by their grades in ascending order. It performs the sort 5 times and returns the average time taken in seconds.

Parameters

studentai	The list of Studentas objects to sort.

Returns

The average time in seconds to sort the students by grade.

5.13.1.6 rikiavimas_pagal_varda_laikas_list()

```
double rikiavimas_pagal_varda_laikas_list (
            list< Studentas > studentai)
```

Measures the time to sort students by name in ascending order.

Measures the time required to sort students by name in ascending order.

This function measures the time it takes to sort the list of students by their names in ascending order. It performs the sort 5 times and returns the average time taken in seconds.

Parameters

studentai	The list of Studentas objects to sort.

Returns

The average time in seconds to sort the students by name.

5.13.1.7 skaitymas_is_failo_list()

```
list< Studentas > skaitymas_is_failo_list (
            list< Studentas > & studentai,
            string failo_pav)
```

Reads student data from a file into a list of Studentas objects.

Reads student data from a file into a `std::list`.

This function opens a file, reads student data (name, surname, and grades), and stores them in a `list<Studentas>`. It also calculates the average and median grades for each student. If the file cannot be opened, it throws an exception.

Parameters

studentai	The list of Studentas objects to store the read data.
failo_pav	The name of the file to read.

Returns

The updated list of Studentas objects.

5.13.1.8 studentu_isskirstymas_list()

```
void studentu_isskirstymas_list (
            string failo_pavadinimas)
```

Manages the entire student data processing workflow.

Processes and segregates students into "vargsiukai" and "galvociai" categories, then writes the results to separate files.

This function reads student data from a file, sorts the students by a chosen criterion, filters them into "vargsiukai" (students with low grades) and "galvociai" (students with good grades), and writes the results into separate files.

Parameters

failo_pavadinimas	The name of the input file to read the data from.

5.13.1.9 vargsiuku_atrinkimas_naudojant_list()

```
list< Studentas > vargsiuku_atrinkimas_naudojant_list (
            list< Studentas > & studentai)
```

Filters students with a final grade below 5.

Filters students into a list of "vargsiukai" (students with low grades).

This function creates a list of students who have a final grade lower than 5.

Parameters

studentai	The list of Studentas objects to filter.

Returns

A new list of students with a final grade less than 5.

5.14 v0_3_header.h File Reference

Function declarations for processing student data using `std::list`.

```
#include "Header.h"
#include <stdio.h>
#include "Studentas.h"
#include "Failas.h"
```

Functions

- list< Studentas > vargsiuku_atrinkimas_naudojant_list (list< Studentas > &studentai)

  Filters students into a list of "vargsiukai" (students with low grades).
- list< Studentas > galvociu_atrinkimas_naudojant_list (list< Studentas > &studentai)

  Filters students into a list of "galvociai" (students with high grades).
- void irasymas (list< Studentas > &studentai, string failo_pav)

  Writes student data to a file.
- void laiko_skaiciavimas_list_konteineris (string failo_pavadinimas, int rikiavimo_pasirinkimas)

  Measures execution times for list operations, including sorting and filtering.
- void irasymas_naudojant_list (list< Studentas > &studentai, string failo_pav)

  Writes student data to a file using a `std::list`.
- list< Studentas > skaitymas_is_failo_list (list< Studentas > &studentai, string failo_pav)

  Reads student data from a file into a `std::list`.
- double rikiavimas_pagal_varda_laikas_list (list< Studentas > studentai)

  Measures the time required to sort students by name in ascending order.
- double rikiavimas_pagal_pavarde_laikas_list (list< Studentas > studentai)

  Measures the time required to sort students by surname in ascending order.
- double rikiavimas_pagal_pazymius_laikas_list (list< Studentas > studentai)

  Measures the time required to sort students by grades in ascending order.
- void studentu_isskirstymas_list (string failo_pavadinimas)

  Processes and segregates students into "vargsiukai" and "galvociai" categories, then writes the results to separate files.

5.14.1 Detailed Description

Function declarations for processing student data using `std::list`.

5.14.2 Function Documentation

5.14.2.1 galvociu_atrinkimas_naudojant_list()

```
list< Studentas > galvociu_atrinkimas_naudojant_list (
            list< Studentas > & studentai)
```

Filters students into a list of "galvociai" (students with high grades).

Parameters

studentai	The list of students to filter.

Returns

A list of "galvociai" students.

Filters students into a list of "galvociai" (students with high grades).

This function creates a list of students who have a final grade of 5 or higher.

Parameters

studentai	The list of Studentas objects to filter.

Returns

A new list of students with a final grade of 5 or higher.

5.14.2.2 irasymas()

```
void irasymas (
            list< Studentas > & studentai,
            string failo_pav)
```

Writes student data to a file.

Parameters

studentai	The list of students to write.
failo_pav	The name of the file (excluding the extension).

5.14.2.3 irasymas_naudojant_list()

```
void irasymas_naudojant_list (
            list< Studentas > & studentai,
            string failo_pav)
```

Writes student data to a file using a `std::list`.

Parameters

studentai	The list of students to write.
failo_pav	The name of the file (excluding the extension).

Writes student data to a file using a `std::list`.

This function saves the filtered lists of students (those with final grades below 5 and those with final grades of 5 or higher) to files.

Parameters

studentai	The list of Studentas objects to write.
failo_pav	The name of the file to write the data to.

5.14.2.4 laiko_skaiciavimas_list_konteineris()

```
void laiko_skaiciavimas_list_konteineris (
            string failo_pavadinimas,
            int rikiavimo_pasirinkimas)
```

Measures execution times for list operations, including sorting and filtering.

Parameters

failo_pavadinimas	The name of the file containing student data (excluding the extension).
rikiavimo_pasirinkimas	The sorting criteria (1 for name, 2 for surname, 3 for grades).

Measures execution times for list operations, including sorting and filtering.

This function calculates and displays the time taken to read the student data from a file, sort them, filter them, and write the results into separate files.

Parameters

failo_pavadinimas	The name of the input file.
rikiavimo_pasirinkimas	The sorting criterion (1 for name, 2 for surname, 3 for grade).

5.14.2.5 rikiavimas_pagal_pavarde_laikas_list()

```
double rikiavimas_pagal_pavarde_laikas_list (
            list< Studentas > studentai)
```

Measures the time required to sort students by surname in ascending order.

Parameters

studentai	The list of students to sort.

Returns

The average sorting time in seconds.

Measures the time required to sort students by surname in ascending order.

This function measures the time it takes to sort the list of students by their surnames in ascending order. It performs the sort 5 times and returns the average time taken in seconds.

Parameters

studentai	The list of Studentas objects to sort.

Returns

The average time in seconds to sort the students by surname.

5.14.2.6 rikiavimas_pagal_pazymius_laikas_list()

```
double rikiavimas_pagal_pazymius_laikas_list (
            list< Studentas > studentai)
```

Measures the time required to sort students by grades in ascending order.

Parameters

studentai	The list of students to sort.

Returns

The average sorting time in seconds.

Measures the time required to sort students by grades in ascending order.

This function measures the time it takes to sort the list of students by their grades in ascending order. It performs the sort 5 times and returns the average time taken in seconds.

Parameters

studentai	The list of Studentas objects to sort.

Returns

The average time in seconds to sort the students by grade.

5.14.2.7 rikiavimas_pagal_varda_laikas_list()

```
double rikiavimas_pagal_varda_laikas_list (
            list< Studentas > studentai)
```

Measures the time required to sort students by name in ascending order.

Parameters

studentai	The list of students to sort.

Returns

The average sorting time in seconds.

Measures the time required to sort students by name in ascending order.

This function measures the time it takes to sort the list of students by their names in ascending order. It performs the sort 5 times and returns the average time taken in seconds.

Parameters

studentai	The list of Studentas objects to sort.

Returns

The average time in seconds to sort the students by name.

5.14.2.8 skaitymas_is_failo_list()

```
list< Studentas > skaitymas_is_failo_list (
            list< Studentas > & studentai,
            string failo_pav)
```

Reads student data from a file into a `std::list`.

Parameters

studentai	The list to populate with student data.
failo_pav	The name of the file (excluding the extension).

Returns

The populated list of students.

Reads student data from a file into a `std::list`.

This function opens a file, reads student data (name, surname, and grades), and stores them in a `list<Studentas>`. It also calculates the average and median grades for each student. If the file cannot be opened, it throws an exception.

Parameters

studentai	The list of Studentas objects to store the read data.
failo_pav	The name of the file to read.

Returns

The updated list of Studentas objects.

5.14.2.9 studentu_isskirstymas_list()

```
void studentu_isskirstymas_list (
            string failo_pavadinimas)
```

Processes and segregates students into "vargsiukai" and "galvociai" categories, then writes the results to separate files.

Parameters

failo_pavadinimas	The name of the file containing student data (excluding the extension).

Processes and segregates students into "vargsiukai" and "galvociai" categories, then writes the results to separate files.

This function reads student data from a file, sorts the students by a chosen criterion, filters them into "vargsiukai" (students with low grades) and "galvociai" (students with good grades), and writes the results into separate files.

Parameters

failo_pavadinimas	The name of the input file to read the data from.

5.14.2.10 vargsiuku_atrinkimas_naudojant_list()

```
list< Studentas > vargsiuku_atrinkimas_naudojant_list (
            list< Studentas > & studentai)
```

Filters students into a list of "vargsiukai" (students with low grades).

Parameters

studentai	The list of students to filter.

Returns

A list of "vargsiukai" students.

Filters students into a list of "vargsiukai" (students with low grades).

This function creates a list of students who have a final grade lower than 5.

Parameters

studentai	The list of Studentas objects to filter.

Returns

A new list of students with a final grade less than 5.

5.15 v0_3_header.h

Go to the documentation of this file.

```
00001
00005 #ifndef v0_3_header_h
00006 #define v0_3_header_h
00007 #include "Header.h"
00008 #include <stdio.h>
00009 #include "Studentas.h"
00010 #include "Failas.h"
00011
00012
00013
00020 list<Studentas> vargsiuku_atrinkimas_naudojant_list(list<Studentas>& studentai);
00021
00028 list<Studentas> galvociu_atrinkimas_naudojant_list(list<Studentas>& studentai);
00029
00036 void irasymas(list<Studentas>& studentai, string failo_pav);
00037
00044 void laiko_skaiciavimas_list_konteineris(string failo_pavadinimas, int rikiavimo_pasirinkimas);
00045
00052 void irasymas_naudojant_list(list<Studentas>& studentai, string failo_pav);
00053
00061 list<Studentas> skaitymas_is_failo_list(list<Studentas>& studentai, string failo_pav);
00062
00069 double rikiavimas_pagal_varda_laikas_list(list<Studentas> studentai);
00070
00077 double rikiavimas_pagal_pavarde_laikas_list(list<Studentas> studentai);
00078
00085 double rikiavimas_pagal_pazymius_laikas_list(list<Studentas> studentai);
00086
00093 void studentu_isskirstymas_list(string failo_pavadinimas);
00094
00095 #endif /* v0_3_header_h */
```

5.16 Zmogus.cpp File Reference

Implementation of the Zmogus class, representing a basic person entity.

```
#include <stdio.h>
#include "Zmogus.h"
#include "Header.h"
```

5.16.1 Detailed Description

Implementation of the Zmogus class, representing a basic person entity.

5.17 Zmogus.h File Reference

Declaration of the Zmogus class, a base class for representing a person entity.

```
#include "Header.h"
```

Classes

- class Zmogus

  Represents a basic person entity with a name and surname.

5.17.1 Detailed Description

Declaration of the `Zmogus` class, a base class for representing a person entity.

5.18 Zmogus.h

Go to the documentation of this file.

```
00001
00005 #ifndef Zmogus_h
00006 #define Zmogus_h
00007 #include "Header.h"
00008
00009
00018 class Zmogus {
00019 protected:
00023     string vardas;
00024
00028     string pavarde;
00029
00030 public:
00034     Zmogus();
00035
00041     Zmogus(string vardas, string pavarde);
00042
00046     virtual ~Zmogus() = default;
00047
00048
00053     virtual const std::string& getVardas() const = 0;
00054
00059     virtual const std::string& getPavarde() const = 0;
00060
00061 };
00062
00063 #endif /* Zmogus_h */
```

Index

~Studentas
    Studentas, 9
~Zmogus
    Zmogus, 16

Failas.cpp, 19
    galvociu_atrinkimas, 20
    irasymas_i_faila, 20
    laiko_skaiciavimas, 20
    laiko_skaiciavimas_failo_generavimas, 21
    pagrindinio_failo_generavimas, 21
    rikiavimas_pagal_pavarde_laikas, 21
    rikiavimas_pagal_pazymius_laikas, 22
    rikiavimas_pagal_varda_laikas, 22
    studentu_isskirstymas, 22
    vargsiuku_atrinkimas, 23
Failas.h, 23
    galvociu_atrinkimas, 24
    irasymas_i_faila, 24
    laiko_skaiciavimas, 25
    laiko_skaiciavimas_failo_generavimas, 25
    pagrindinio_failo_generavimas, 26
    rikiavimas_pagal_pavarde_laikas, 26
    rikiavimas_pagal_pazymius_laikas, 27
    rikiavimas_pagal_varda_laikas, 27
    studentu_isskirstymas, 28
    vargsiuku_atrinkimas, 28

galvociu_atrinkimas
    Failas.cpp, 20
    Failas.h, 24
galvociu_atrinkimas_naudojant_list
    v0_3.cpp, 52
    v0_3_header.h, 56
getEgzaminas
    Studentas, 10
getGalutinisMed
    Studentas, 10
getGalutinisVid
    Studentas, 10
getNamuDarbai
    Studentas, 10
getPavarde
    Studentas, 11
    Zmogus, 16
getVardas
    Studentas, 11
    Zmogus, 16

Header.h, 29

irasymas
    Studentas.cpp, 38
    Studentas.h, 44
    v0_3_header.h, 57
irasymas_i_faila
    Failas.cpp, 20
    Failas.h, 24
irasymas_list
    Studentas.cpp, 38
    Studentas.h, 44
irasymas_naudojant_list
    v0_3.cpp, 52
    v0_3_header.h, 57
isvedimas
    Studentas.cpp, 38
    Studentas.h, 45
isvedimas_list
    Studentas.cpp, 39
    Studentas.h, 45
isvedimas_su_mediana
    Studentas.cpp, 39
    Studentas.h, 46
isvedimas_su_vidurkiu
    Studentas.cpp, 39
    Studentas.h, 46
ivedimas
    Studentas.cpp, 40
    Studentas.h, 46
ivesties_isvesties_metodu_demonstracija
    Studentas.cpp, 40
    Studentas.h, 47

laiko_skaiciavimas
    Failas.cpp, 20
    Failas.h, 25
laiko_skaiciavimas_failo_generavimas
    Failas.cpp, 21
    Failas.h, 25
laiko_skaiciavimas_list_konteineris
    v0_3.cpp, 53
    v0_3_header.h, 58

main
    main.cpp, 31
main.cpp, 30
    main, 31
medianosSkaiciavimas
    Studentas, 11

operator<<

Studentas, 13
Studentas.cpp, 40
operator>>
Studentas, 13
Studentas.cpp, 40
operator=
Studentas, 11
operator==
strategijos.cpp, 31
strategijos.h, 34

pagrindinio_failo_generavimas
Failas.cpp, 21
Failas.h, 26
palyginti_pavardes
Studentas.cpp, 41
Studentas.h, 47
palyginti_pazymius
Studentas.cpp, 41
Studentas.h, 47
palyginti_vardus
Studentas.cpp, 41
Studentas.h, 48
pavarde
Zmogus, 17

random_egz
Studentas.cpp, 42
Studentas.h, 48
random_pazymiai
Studentas.cpp, 42
Studentas.h, 49
rikiavimas_pagal_pavarde_laikas
Failas.cpp, 21
Failas.h, 26
rikiavimas_pagal_pavarde_laikas_list
v0_3.cpp, 53
v0_3_header.h, 58
rikiavimas_pagal_pazymius_laikas
Failas.cpp, 22
Failas.h, 27
rikiavimas_pagal_pazymius_laikas_list
v0_3.cpp, 53
v0_3_header.h, 59
rikiavimas_pagal_varda_laikas
Failas.cpp, 22
Failas.h, 27
rikiavimas_pagal_varda_laikas_list
v0_3.cpp, 54
v0_3_header.h, 59
rule_of_three_metodu_demonstracija
Studentas.cpp, 42
Studentas.h, 49

setEgzaminas
Studentas, 12
setNamuDarbai
Studentas, 12
setPavarde
Studentas, 12
setVardas
Studentas, 13
skaidymas_2_strategija_list
strategijos.cpp, 32
strategijos.h, 34
skaidymas_2_strategija_vector
strategijos.cpp, 32
strategijos.h, 35
skaidymas_3_strategija_list
strategijos.cpp, 32
strategijos.h, 35
skaidymas_3_strategija_vector
strategijos.cpp, 33
strategijos.h, 36
skaitymas_is_failo
Studentas.cpp, 42
Studentas.h, 49
skaitymas_is_failo_list
v0_3.cpp, 54
v0_3_header.h, 60
strategijos.cpp, 31
operator==, 31
skaidymas_2_strategija_list, 32
skaidymas_2_strategija_vector, 32
skaidymas_3_strategija_list, 32
skaidymas_3_strategija_vector, 33
strategijos.h, 33
operator==, 34
skaidymas_2_strategija_list, 34
skaidymas_2_strategija_vector, 35
skaidymas_3_strategija_list, 35
skaidymas_3_strategija_vector, 36
Studentas, 7
~Studentas, 9
getEgzaminas, 10
getGalutinisMed, 10
getGalutinisVid, 10
getNamuDarbai, 10
getPavarde, 11
getVardas, 11
medianosSkaiciavimas, 11
operator<<, 13
operator>>, 13
operator=, 11
setEgzaminas, 12
setNamuDarbai, 12
setPavarde, 12
setVardas, 13
Studentas, 9
vidurkioSkaiciavimas, 13
Studentas.cpp, 37
irasymas, 38
irasymas_list, 38
isvedimas, 38
isvedimas_list, 39
isvedimas_su_mediana, 39
isvedimas_su_vidurkiu, 39

ivedimas, 40
ivesties_isvesties_metodu_demonstracija, 40
operator<<, 40
operator>>, 40
palyginti_pavardes, 41
palyginti_pazymius, 41
palyginti_vardus, 41
random_egz, 42
random_pazymiai, 42
rule_of_three_metodu_demonstracija, 42
skaitymas_is_failo, 42
valymas, 43
Studentas.h, 43
irasymas, 44
irasymas_list, 44
isvedimas, 45
isvedimas_list, 45
isvedimas_su_mediana, 46
isvedimas_su_vidurkiu, 46
ivedimas, 46
ivesties_isvesties_metodu_demonstracija, 47
palyginti_pavardes, 47
palyginti_pazymius, 47
palyginti_vardus, 48
random_egz, 48
random_pazymiai, 49
rule_of_three_metodu_demonstracija, 49
skaitymas_is_failo, 49
valymas, 50
studentu_isskirstymas
Failas.cpp, 22
Failas.h, 28
studentu_isskirstymas_list
v0_3.cpp, 55
v0_3_header.h, 60

v0_3.cpp, 52
galvociu_atrinkimas_naudojant_list, 52
irasymas_naudojant_list, 52
laiko_skaiciavimas_list_konteineris, 53
rikiavimas_pagal_pavarde_laikas_list, 53
rikiavimas_pagal_pazymius_laikas_list, 53
rikiavimas_pagal_varda_laikas_list, 54
skaitymas_is_failo_list, 54
studentu_isskirstymas_list, 55
vargsiuku_atrinkimas_naudojant_list, 55
v0_3_header.h, 56
galvociu_atrinkimas_naudojant_list, 56
irasymas, 57
irasymas_naudojant_list, 57
laiko_skaiciavimas_list_konteineris, 58
rikiavimas_pagal_pavarde_laikas_list, 58
rikiavimas_pagal_pazymius_laikas_list, 59
rikiavimas_pagal_varda_laikas_list, 59
skaitymas_is_failo_list, 60
studentu_isskirstymas_list, 60
vargsiuku_atrinkimas_naudojant_list, 61
valymas
Studentas.cpp, 43
Studentas.h, 50
vardas
Zmogus, 17
vargsiuku_atrinkimas
Failas.cpp, 23
Failas.h, 28
vargsiuku_atrinkimas_naudojant_list
v0_3.cpp, 55
v0_3_header.h, 61
vidurkioSkaiciavimas
Studentas, 13

Zmogus, 14
∼Zmogus, 16
getPavarde, 16
getVardas, 16
pavarde, 17
vardas, 17
Zmogus, 15
Zmogus.cpp, 62
Zmogus.h, 62