Student Management System

Generated by Doxygen 1.13.2

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 Laikas Class Reference

Class for measuring execution time.

```
#include <laikas.h>
```

Public Member Functions

- Laikas (const std::string &pavadinimas)

  Constructor.
- void pradeti ()

  Start the timer Records the starting time point.
- void baigti ()

  Stop the timer and print the elapsed time Records the ending time point and outputs the time difference.
- double gautiLaikoSkirtuma ()

  Get the elapsed time in seconds.

Private Attributes

- std::chrono::high_resolution_clock::time_point start

  Start time point.
- std::chrono::high_resolution_clock::time_point end

  End time point.
- std::string veiksmoPavadinimas

  Name of the operation being timed.

4.1.1 Detailed Description

Class for measuring execution time.

This class provides functionality to measure and report the execution time of code segments using high-resolution clock.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Laikas()

```
Laikas::Laikas (
            const std::string & pavadinimas)
```

Constructor.

Parameters

pavadinimas	Name of the operation to time

4.1.3 Member Function Documentation

4.1.3.1 baigti()

```
void Laikas::baigti ()
```

Stop the timer and print the elapsed time Records the ending time point and outputs the time difference.

4.1.3.2 gautiLaikoSkirtuma()

```
double Laikas::gautiLaikoSkirtuma ()
```

Get the elapsed time in seconds.

Returns

Time difference between start and end in seconds

4.1.3.3 pradeti()

```
void Laikas::pradeti ()
```

Start the timer Records the starting time point.

4.1.4 Member Data Documentation

4.1.4.1 end

```
std::chrono::high_resolution_clock::time_point Laikas::end  [private]
```

End time point.

4.1.4.2 start

```
std::chrono::high_resolution_clock::time_point Laikas::start  [private]
```

Start time point.

4.1.4.3 veiksmoPavadinimas

```
std::string Laikas::veiksmoPavadinimas  [private]
```

Name of the operation being timed.

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

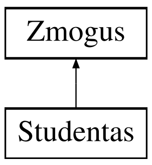
- laikas.h
- laikas.cpp

4.2 Studentas Class Reference

Student class that inherits from Zmogus (Person) abstract class.

```
#include <funkcijos.h>
```

Inheritance diagram for Studentas:



Public Member Functions

- Studentas ()

  Default constructor Initializes a student with default values.
- Studentas (std::istream &is)

  Constructor that initializes a student from input stream.
- Studentas (const std::string &vardas, const std::string &pavarde)

  Constructor with name and surname.
- Studentas (const Studentas &other)

  Copy constructor.
- Studentas & operator= (const Studentas &other)

  Copy assignment operator.
- Studentas (Studentas &&other) noexcept

  Move constructor.
- Studentas & operator= (Studentas &&other) noexcept

  Move assignment operator.
- ∼Studentas () override

  Virtual destructor Overrides the pure virtual destructor from Zmogus.
- void print (std::ostream &os) const override

  Prints student information to output stream.
- void read (std::istream &is) override

  Reads student information from input stream.
- std::vector< int > nd () const

  Get homework grades.

- int egzaminas () const

  Get exam grade.
- double galutinis () const

  Get final grade.
- void setEgzaminas (int egzaminas)

  Set exam grade.
- void setGalutinis (double galutinis)

  Set final grade.
- std::istream & readStudent (std::istream &is)

  Read student data from input stream.
- void addND (int pazymys)

  Add a homework grade.
- void clearND ()

  Clear all homework grades.
- double skaiciuotiVid () const

  Calculate average of homework grades.
- double skaiciuotiMed () const

  Calculate median of homework grades.
- double galBalas (bool naudotiVidurki=true) const

  Calculate final grade.

Public Member Functions inherited from Zmogus

- Zmogus ()=default

  Default constructor.
- Zmogus (const std::string &vardas, const std::string &pavarde)

  Constructor with name and surname.
- Zmogus (const Zmogus &other)

  Copy constructor.
- Zmogus & operator= (const Zmogus &other)

  Copy assignment operator.
- Zmogus (Zmogus &&other) noexcept

  Move constructor.
- Zmogus & operator= (Zmogus &&other) noexcept

  Move assignment operator.
- virtual ∼Zmogus ()=0

  Virtual destructor Pure virtual destructor makes this class abstract.
- std::string vardas () const

  Get first name.
- std::string pavarde () const

  Get last name.
- void setVardas (const std::string &vardas)

  Set first name.
- void setPavarde (const std::string &pavarde)

  Set last name.

Static Public Member Functions

- static void nuskaitymasFile (std::vector< Studentas > &grupe, const std::string &failoPavadinimas)

  Static method for reading students from file.

Static Public Attributes

- static int destruktoriuSk = 0

  Static counter for tracking destructor calls (for testing)

Private Attributes

- std::vector< int > nd_

  Vector of homework (namų darbų) scores.
- int egzaminas_

  Exam score.
- double galutinis_

  Final grade.

Additional Inherited Members

Protected Attributes inherited from Zmogus

- std::string vardas_

  First name of the person.
- std::string pavarde_

  Last name of the person.

4.2.1 Detailed Description

Student class that inherits from Zmogus (Person) abstract class.

This class represents a student with homework grades, exam score and final grade. It inherits from the abstract Zmogus class and implements its pure virtual methods.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Studentas() `[1/5]`

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

Default constructor Initializes a student with default values.

4.2.2.2 Studentas() `[2/5]`

```
Studentas::Studentas (
              std::istream & is)
```

Constructor that initializes a student from input stream.

Parameters

is	Input stream to read from

4.2.2.3 Studentas() `[3/5]`

```
Studentas::Studentas (
            const std::string & vardas,
            const std::string & pavarde)  [inline]
```

Constructor with name and surname.

Parameters

vardas	Student's first name
pavarde	Student's last name

4.2.2.4 Studentas() `[4/5]`

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

Copy constructor.

Parameters

other	Another student to copy from

4.2.2.5 Studentas() `[5/5]`

```
Studentas::Studentas (
            Studentas && other)  [noexcept]
```

Move constructor.

Parameters

other	Another student to move from

4.2.2.6 ∼Studentas()

```
Studentas::∼Studentas ()  [override]
```

Virtual destructor Overrides the pure virtual destructor from Zmogus.

4.2.3 Member Function Documentation

4.2.3.1 addND()

```
void Studentas::addND (
            int pazymys)
```

Add a homework grade.

Parameters

pazymys	Grade to add

Exceptions

std::invalid_argument	If grade is not between 1 and 10

4.2.3.2 clearND()

```
void Studentas::clearND ()
```

Clear all homework grades.

4.2.3.3 egzaminas()

```
int Studentas::egzaminas () const  [inline]
```

Get exam grade.

Returns

Exam grade

4.2.3.4 galBalas()

```
double Studentas::galBalas (
             bool naudotiVidurki = true) const
```

Calculate final grade.

Parameters

naudotiVidurki	Whether to use average (true) or median (false)

Returns

Final grade calculated as 0.4∗homework + 0.6∗exam

4.2.3.5 galutinis()

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

Get final grade.

Returns

Final grade

4.2.3.6 nd()

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

Get homework grades.

Returns

Vector of homework grades

4.2.3.7 nuskaitymasFile()

```
void Studentas::nuskaitymasFile (
             std::vector< Studentas > & grupe,
             const std::string & failoPavadinimas)  [static]
```

Static method for reading students from file.

Parameters

grupe	Vector to store students
failoPavadinimas	Filename to read from

4.2.3.8 operator=() [1/2]

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

Copy assignment operator.

Parameters

other	Another student to copy from

Returns

Reference to this student after assignment

4.2.3.9 operator=() [2/2]

```
Studentas & Studentas::operator= (
             Studentas && other)  [noexcept]
```

Move assignment operator.

Parameters

other	Another student to move from

Returns

Reference to this student after assignment

4.2.3.10 print()

```
void Studentas::print (
            std::ostream & os) const  [override], [virtual]
```

Prints student information to output stream.

Parameters

os	Output stream to print to

Implements Zmogus.

4.2.3.11 read()

```
void Studentas::read (
             std::istream & is)  [override], [virtual]
```

Reads student information from input stream.

Parameters

is	Input stream to read from

Implements Zmogus.

4.2.3.12 readStudent()

```
std::istream & Studentas::readStudent (
             std::istream & is)
```

Read student data from input stream.

Parameters

is	Input stream to read from

Returns

Reference to the input stream

4.2.3.13 setEgzaminas()

```
void Studentas::setEgzaminas (
             int egzaminas)  [inline]
```

Set exam grade.

Parameters

egzaminas	Exam grade to set

4.2.3.14 setGalutinis()

```
void Studentas::setGalutinis (
             double galutinis)  [inline]
```

Set final grade.

Parameters

galutinis	Final grade to set

4.2.3.15 skaiciuotiMed()

```
double Studentas::skaiciuotiMed () const
```

Calculate median of homework grades.

Returns

Median of homework grades

Exceptions

std::runtime_error	If there are no homework grades

4.2.3.16 skaiciuotiVid()

```
double Studentas::skaiciuotiVid () const
```

Calculate average of homework grades.

Returns

Average of homework grades

Exceptions

std::runtime_error	If there are no homework grades

4.2.4 Member Data Documentation

4.2.4.1 destruktoriuSk

```
int Studentas::destruktoriuSk = 0  [static]
```

Static counter for tracking destructor calls (for testing)

4.2.4.2 egzaminas_

```
int Studentas::egzaminas_  [private]
```

Exam score.

4.2.4.3 galutinis_

```
double Studentas::galutinis_  [private]
```

Final grade.

4.2.4.4 nd_

```
std::vector<int> Studentas::nd_  [private]
```

Vector of homework (namų darbų) scores.

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

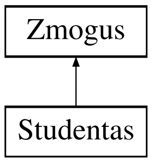
- funkcijos.h
- funkcijos.cpp

4.3 Zmogus Class Reference

Abstract base class representing a person.

```
#include <zmogus.h>
```

Inheritance diagram for Zmogus:



Public Member Functions

- Zmogus ()=default

  Default constructor.
- Zmogus (const std::string &vardas, const std::string &pavarde)

  Constructor with name and surname.
- Zmogus (const Zmogus &other)

  Copy constructor.
- Zmogus & operator= (const Zmogus &other)

  Copy assignment operator.
- Zmogus (Zmogus &&other) noexcept

  Move constructor.
- Zmogus & operator= (Zmogus &&other) noexcept

  Move assignment operator.
- virtual ∼Zmogus ()=0

  Virtual destructor Pure virtual destructor makes this class abstract.
- virtual void print (std::ostream &os) const =0

  Print person information to output stream.

- virtual void read (std::istream &is)=0

  Read person information from input stream.
- std::string vardas () const

  Get first name.
- std::string pavarde () const

  Get last name.
- void setVardas (const std::string &vardas)

  Set first name.
- void setPavarde (const std::string &pavarde)

  Set last name.

Protected Attributes

- std::string vardas_

  First name of the person.
- std::string pavarde_

  Last name of the person.

4.3.1 Detailed Description

Abstract base class representing a person.

This class serves as an abstract base class for all person types. It implements common attributes (name, surname) and declares pure virtual methods.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Zmogus() [1/4]

```
Zmogus::Zmogus ()  [default]
```

Default constructor.

4.3.2.2 Zmogus() [2/4]

```
Zmogus::Zmogus (
            const std::string & vardas,
            const std::string & pavarde)  [inline]
```

Constructor with name and surname.

Parameters

vardas	First name
pavarde	Last name

4.3.2.3 Zmogus() [3/4]

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

Copy constructor.

Parameters

other	Another person to copy from

4.3.2.4 Zmogus() [4/4]

```
Zmogus::Zmogus (
            Zmogus && other)  [noexcept]
```

Move constructor.

Parameters

other	Another person to move from

4.3.2.5 ∼Zmogus()

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

Virtual destructor Pure virtual destructor makes this class abstract.

4.3.3 Member Function Documentation

4.3.3.1 operator=() [1/2]

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

Copy assignment operator.

Parameters

other	Another person to copy from

Returns

Reference to this person after assignment

4.3.3.2 operator=() [2/2]

```
Zmogus & Zmogus::operator= (
            Zmogus && other)  [noexcept]
```

Move assignment operator.

Parameters

other	Another person to move from

Returns

Reference to this person after assignment

4.3.3.3 pavarde()

```
std::string Zmogus::pavarde () const  [inline]
```

Get last name.

Returns

Last name of the person

4.3.3.4 print()

```
virtual void Zmogus::print (
             std::ostream & os) const  [pure virtual]
```

Print person information to output stream.

Parameters

os	Output stream to print to

Pure virtual method that derived classes must implement

Implemented in Studentas.

4.3.3.5 read()

```
virtual void Zmogus::read (
             std::istream & is)  [pure virtual]
```

Read person information from input stream.

Parameters

is	Input stream to read from

Pure virtual method that derived classes must implement

Implemented in Studentas.

4.3.3.6 setPavarde()

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

Set last name.

Parameters

pavarde	Last name to set

4.3.3.7 setVardas()

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

Set first name.

Parameters

vardas	First name to set

4.3.3.8 vardas()

```
std::string Zmogus::vardas () const  [inline]
```

Get first name.

Returns

First name of the person

4.3.4 Member Data Documentation

4.3.4.1 pavarde_

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

Last name of the person.

4.3.4.2 vardas_

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

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 funkcijos.cpp File Reference

```
#include "funkcijos.h"
#include "laikas.h"
```

Functions

- bool compareByVardas (const Studentas &a, const Studentas &b)

  Compare students by first name.
- bool compareByPavarde (const Studentas &a, const Studentas &b)

  Compare students by last name.
- bool compareByGalutinis (const Studentas &a, const Studentas &b)

  Compare students by final grade (descending)
- void skaitytiIsFailo (std::vector< Studentas > &grupe, const std::string &failoPavadinimas)

  Read students from file.
- void isvestiStudentusIFaila (const std::vector< Studentas > &studentai, const std::string &failoPavadinimas, char ats)

  Write students to file.
- void sortStudentai (std::vector< Studentas > &grupe, char sortingOption)

  Sort students by specified criterion.
- void skirstytiStudentus (std::vector< Studentas > &grupe, std::vector< Studentas > &kietiakiai, std::vector< Studentas > &vargsai)

  Split students into two groups based on final grade.
- void testuotiDuomenuApdorojima (const std::string &aplankas, int skaicius)

  Test data processing performance.
- void testuotiStudentoMetodus ()

  Test Studentas class methods.
- void testuotiZmogausKlase ()

  Test Zmogus class.

5.1.1 Function Documentation

5.1.1.1 compareByGalutinis()

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

Compare students by final grade (descending)

Parameters

a	First student
b	Second student

Returns

True if a's final grade is higher than b's

5.1.1.2 compareByPavarde()

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

Compare students by last name.

Parameters

a	First student
b	Second student

Returns

True if a's surname comes before b's

5.1.1.3 compareByVardas()

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

Compare students by first name.

Parameters

a	First student
b	Second student

Returns

True if a's name comes before b's

5.1.1.4 isvestiStudentusIFaila()

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

Write students to file.

Parameters

studentai	Students to write
failoPavadinimas	Filename to write to
ats	Whether to use average ('v') or median ('m') for final grade

5.1.1.5 skaitytiIsFailo()

```
void skaitytiIsFailo (
            std::vector< Studentas > & grupe,
            const std::string & failoPavadinimas)
```

Read students from file.

Parameters

grupe	Vector to store students
failoPavadinimas	Filename to read from

5.1.1.6 skirstytiStudentus()

```
void skirstytiStudentus (
            std::vector< Studentas > & grupe,
            std::vector< Studentas > & kietiakiai,
            std::vector< Studentas > & vargsai)
```

Split students into two groups based on final grade.

Parameters

grupe	Input vector of students
kietiakiai	Output vector for students with final grade >= 5.0
vargsai	Output vector for students with final grade < 5.0

5.1.1.7 sortStudentai()

```
void sortStudentai (
            std::vector< Studentas > & grupe,
            char sortingOption)
```

Sort students by specified criterion.

Parameters

grupe	Students to sort
sortingOption	Sorting criterion: 'v' (name), 'p' (surname), 'g' (final grade)

5.1.1.8 testuotiDuomenuApdorojima()

```
void testuotiDuomenuApdorojima (
            const std::string & aplankas,
            int skaicius)
```

Test data processing performance.

Parameters

aplankas	Directory for files
skaicius	Number of students

5.1.1.9 testuotiStudentoMetodus()

```
void testuotiStudentoMetodus ()
```

Test Studentas class methods.

5.1.1.10 testuotiZmogausKlase()

```
void testuotiZmogausKlase ()
```

Test Zmogus class.

5.2 funkcijos.h File Reference

```
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <iomanip>
#include <fstream>
#include <sstream>
#include <stdexcept>
#include <execution>
#include <numeric>
#include <cassert>
#include "zmogus.h"
```

Classes

- class Studentas

    Student class that inherits from Zmogus (Person) abstract class.

Functions

- bool compareByVardas (const Studentas &a, const Studentas &b)

  Compare students by first name.
- bool compareByPavarde (const Studentas &a, const Studentas &b)

  Compare students by last name.
- bool compareByGalutinis (const Studentas &a, const Studentas &b)

  Compare students by final grade (descending)
- void skaitytiIsFailo (std::vector< Studentas > &grupe, const std::string &failoPavadinimas)

  Read students from file.
- void isvestiStudentusIFaila (const std::vector< Studentas > &studentai, const std::string &failoPavadinimas, char ats)

  Write students to file.
- void sortStudentai (std::vector< Studentas > &grupe, char sortingOption)

  Sort students by specified criterion.
- void skirstytiStudentus (std::vector< Studentas > &grupe, std::vector< Studentas > &kietiakiai, std::vector< Studentas > &vargsai)

  Split students into two groups based on final grade.
- void testuotiDuomenuApdorojima (const std::string &aplankas, int skaicius)

  Test data processing performance.
- void testuotiStudentoMetodus ()

  Test Studentas class methods.
- void testuotiZmogausKlase ()

  Test Zmogus class.

5.2.1 Function Documentation

5.2.1.1 compareByGalutinis()

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

Compare students by final grade (descending)

Parameters

a	First student
b	Second student

Returns

True if a's final grade is higher than b's

5.2.1.2 compareByPavarde()

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

Compare students by last name.

Parameters

a	First student
b	Second student

Returns

True if a's surname comes before b's

5.2.1.3 compareByVardas()

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

Compare students by first name.

Parameters

a	First student
b	Second student

Returns

True if a's name comes before b's

5.2.1.4 isvestiStudentusIFaila()

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

Write students to file.

Parameters

studentai	Students to write
failoPavadinimas	Filename to write to
ats	Whether to use average ('v') or median ('m') for final grade

5.2.1.5 skaitytiIsFailo()

```
void skaitytiIsFailo (
            std::vector< Studentas > & grupe,
            const std::string & failoPavadinimas)
```

Read students from file.

Parameters

grupe	Vector to store students
failoPavadinimas	Filename to read from

5.2.1.6 skirstytiStudentus()

```
void skirstytiStudentus (
            std::vector< Studentas > & grupe,
            std::vector< Studentas > & kietiakiai,
            std::vector< Studentas > & vargsai)
```

Split students into two groups based on final grade.

Parameters

grupe	Input vector of students
kietiakiai	Output vector for students with final grade >= 5.0
vargsai	Output vector for students with final grade < 5.0

5.2.1.7 sortStudentai()

```
void sortStudentai (
            std::vector< Studentas > & grupe,
            char sortingOption)
```

Sort students by specified criterion.

Parameters

grupe	Students to sort
sortingOption	Sorting criterion: 'v' (name), 'p' (surname), 'g' (final grade)

5.2.1.8 testuotiDuomenuApdorojima()

```
void testuotiDuomenuApdorojima (
            const std::string & aplankas,
            int skaicius)
```

Test data processing performance.

Parameters

aplankas	Directory for files
skaicius	Number of students

5.2.1.9 testuotiStudentoMetodus()

```
void testuotiStudentoMetodus ()
```

Test Studentas class methods.

5.2.1.10 testuotiZmogausKlase()

```
void testuotiZmogausKlase ()
```

Test Zmogus class.

5.3 funkcijos.h

Go to the documentation of this file.

```
#ifndef FUNKCIJOS_H
#define FUNKCIJOS_H

#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <iomanip>
#include <fstream>
#include <sstream>
#include <stdexcept>
#include <execution>
#include <numeric>
#include <cassert>
#include "zmogus.h"

class Studentas : public Zmogus
{
// realizacija
private:
    std::vector<int> nd_;
    int egzaminas_;
    double galutinis_;

// interfeisas
public:
    static int destruktoriuSk;

    Studentas() : Zmogus(), egzaminas_(0), galutinis_(0) { }

    Studentas(std::istream& is);

    Studentas(const std::string& vardas, const std::string& pavarde)
        : Zmogus(vardas, pavarde), egzaminas_(0), galutinis_(0) { }

    // Rule of Five
    Studentas(const Studentas& other);

    Studentas& operator=(const Studentas& other);

    Studentas(Studentas&& other) noexcept;

    Studentas& operator=(Studentas&& other) noexcept;

    ~Studentas() override;

    // Implementation of pure virtual methods
    void print(std::ostream& os) const override;

    void read(std::istream& is) override;

    // Getteriai
    inline std::vector<int> nd() const { return nd_; }

    inline int egzaminas() const { return egzaminas_; }

    inline double galutinis() const { return galutinis_; }

    // Setteriai
    inline void setEgzaminas(int egzaminas) { egzaminas_ = egzaminas; }

    inline void setGalutinis(double galutinis) { galutinis_ = galutinis; }

    // Metodai
    std::istream& readStudent(std::istream& is);

    void addND(int pazymys);

    void clearND();

    double skaiciuotiVid() const;

    double skaiciuotiMed() const;

    double galBalas(bool naudotiVidurki = true) const;

    static void nuskaitymasFile(std::vector<Studentas>& grupe, const std::string& failoPavadinimas);
};

bool compareByVardas(const Studentas& a, const Studentas& b);

bool compareByPavarde(const Studentas& a, const Studentas& b);

bool compareByGalutinis(const Studentas& a, const Studentas& b);

void skaitytiIsFailo(std::vector<Studentas>& grupe, const std::string& failoPavadinimas);

void isvestiStudentusIFaila(const std::vector<Studentas>& studentai, const std::string&
failoPavadinimas, char ats);

void sortStudentai(std::vector<Studentas>& grupe, char sortingOption);

void skirstytiStudentus(std::vector<Studentas>& grupe, std::vector<Studentas>& kietiakiai,
std::vector<Studentas>& vargsai);

void testuotiDuomenuApdorojima(const std::string& aplankas, int skaicius);

void testuotiStudentoMetodus();

void testuotiZmogausKlase();

#endif // FUNKCIJOS_H
```

5.4 laikas.cpp File Reference

```
#include "Laikas.h"
#include "funkcijos.h"
```

5.5 laikas.h File Reference

```
#include <chrono>
#include <iostream>
```

Classes

- class Laikas

  Class for measuring execution time.

5.6 laikas.h

Go to the documentation of this file.

```
#ifndef LAIKAS_H
#define LAIKAS_H

#include <chrono>
#include <iostream>

class Laikas
{
private:
    std::chrono::high_resolution_clock::time_point start;
    std::chrono::high_resolution_clock::time_point end;
    std::string veiksmoPavadinimas;

public:
    Laikas(const std::string& pavadinimas);

    void pradeti();

    void baigti();

    double gautiLaikoSkirtuma();
};

#endif
```

5.7 main.cpp File Reference

```
#include "funkcijos.h"
#include "laikas.h"
#include "zmogus.h"
```

Functions

- int main ()

5.7.1 Function Documentation

5.7.1.1 main()

```
int main ()
```

5.8 zmogus.cpp File Reference

```
#include "zmogus.h"
```

Functions

- std::ostream & operator<< (std::ostream &os, const Zmogus &zmogus)

  Output operator for Zmogus class.
- std::istream & operator>> (std::istream &is, Zmogus &zmogus)

  Input operator for Zmogus class.

5.8.1 Function Documentation

5.8.1.1 operator<<()

```
std::ostream & operator<< (
            std::ostream & os,
            const Zmogus & zmogus)
```

Output operator for Zmogus class.

Parameters

os	Output stream
zmogus	Person to output

Returns

Reference to output stream

5.8.1.2 operator>>()

```
std::istream & operator>> (
            std::istream & is,
            Zmogus & zmogus)
```

Input operator for Zmogus class.

Parameters

is	Input stream
zmogus	Person to input to

Returns

Reference to input stream

5.9 zmogus.h File Reference

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

Classes

- class Zmogus

  Abstract base class representing a person.

Functions

- std::ostream & operator<< (std::ostream &os, const Zmogus &zmogus)

  Output operator for Zmogus class.
- std::istream & operator>> (std::istream &is, Zmogus &zmogus)

  Input operator for Zmogus class.

5.9.1 Function Documentation

5.9.1.1 operator<<()

```
std::ostream & operator<< (
            std::ostream & os,
            const Zmogus & zmogus)
```

Output operator for Zmogus class.

Parameters

os	Output stream
zmogus	Person to output

Returns

Reference to output stream

5.9.1.2 operator>>()

```
std::istream & operator>> (
            std::istream & is,
            Zmogus & zmogus)
```

Input operator for Zmogus class.

Parameters

is	Input stream
zmogus	Person to input to

Returns

Reference to input stream

5.10 zmogus.h

Go to the documentation of this file.

```
#ifndef ZMOGUS_H
#define ZMOGUS_H

#include <string>
#include <iostream>

class Zmogus
{
protected:
    std::string vardas_;
    std::string pavarde_;

public:
    Zmogus() = default;

    Zmogus(const std::string& vardas, const std::string& pavarde)
        : vardas_(vardas), pavarde_(pavarde) {}

    // Rule of Five
    Zmogus(const Zmogus& other) : vardas_(other.vardas_), pavarde_(other.pavarde_) {}

    Zmogus& operator=(const Zmogus& other);

    Zmogus(Zmogus&& other) noexcept;

    Zmogus& operator=(Zmogus&& other) noexcept;

    virtual ~Zmogus() = 0;

    virtual void print(std::ostream& os) const = 0;

    virtual void read(std::istream& is) = 0;

    // Getters and setters
    inline std::string vardas() const { return vardas_; }

    inline std::string pavarde() const { return pavarde_; }

    inline void setVardas(const std::string& vardas) { vardas_ = vardas; }

    inline void setPavarde(const std::string& pavarde) { pavarde_ = pavarde; }
};

std::ostream& operator«(std::ostream& os, const Zmogus& zmogus);

std::istream& operator»(std::istream& is, Zmogus& zmogus);

#endif // ZMOGUS_H
```

Index