My Project

Generated by Doxygen 1.11.0

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

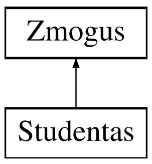
Class Documentation

4.1 Studentas Class Reference

Class representing a student.

```
#include <Studentas.h>
```

Inheritance diagram for Studentas:



Public Member Functions

- Studentas ()

    Default constructor for Studentas class.
- Studentas (std::istream &is)

    Constructor that reads student data from an input stream.
- std::string getVardas () const
- std::string getPavarde () const
- double getEgzaminoRez () const
- std::vector< double > & getNamudarbuRez ()
- void setVardas (const std::string &vardas)
- void setPavarde (const std::string &pavarde)
- void setEgzaminoRez (double egzaminorez)
- void addGrade (double grade)
- std::istream & readStudent (std::istream &)

    Reads student data from an input stream.
- void calculateFinalGrades ()

    Calculates the final grades for the student.
- ~Studentas ()

    ClassDestructor for Studentas class.
- Studentas (const Studentas &other)

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

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

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

  Move assignment operator implementationconstructor implementation.

Public Member Functions inherited from Zmogus

- virtual ∼Zmogus ()=default

Public Attributes

- double namudarburezsuma_
- double vidurkis_
- double galutinisbalasvidurkis_
- double mediana_
- double galutinisbalasmediana_

4.1.1 Detailed Description

Class representing a student.

This class stores information about a student, including their name, grades for homework and exams, and calculated final grades.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Studentas() [1/4]

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

Default constructor for Studentas class.

Default constructor.

Initializes a new Studentas object with default values.

4.1.2.2 Studentas() [2/4]

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

Constructor that reads student data from an input stream.

Constructor to initialize a student with input stream.

Parameters

is	The input stream to read from.

4.1.2.3 ∼Studentas()

```
Studentas::∼Studentas ()
```

ClassDestructor for Studentas class.

Destructor.

Destructor for the Studentas class.

4.1.2.4 Studentas() [3/4]

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

Copy constructor for Studentas class.

Copy constructor.

Constructor for the Studentas class.

4.1.2.5 Studentas() [4/4]

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

Move constructor implementation.

Move constructor.

Move constructor implementation.

4.1.3 Member Function Documentation

4.1.3.1 addGrade()

```
void Studentas::addGrade (
            double grade)   [inline]
```

Add a grade to the homework grades of the student.

4.1.3.2 calculateFinalGrades()

```
void Studentas::calculateFinalGrades ()
```

Calculates the final grades for the student.

Calculate final grades for the student.

This function calculates both the final average grade and the final median grade based on the student's homework and exam results.

4.1.3.3 getEgzaminoRez()

```
double Studentas::getEgzaminoRez () const  [inline], [virtual]
```

Get the exam grade of the student.

Implements Zmogus.

4.1.3.4 getNamudarbuRez()

```
std::vector< double > & Studentas::getNamudarbuRez ()  [inline], [virtual]
```

Get the grades of homework for the student.

Implements Zmogus.

4.1.3.5 getPavarde()

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

Get the last name of the student.

Implements Zmogus.

4.1.3.6 getVardas()

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

Get the first name of the student.

Implements Zmogus.

4.1.3.7 operator=() [1/2]

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

Copy assignment operator implementation.

Copy assignment operator.

Copy assignment operator implementation.

4.1.3.8 operator=() [2/2]

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

Move assignment operator implementationconstructor implementation.

Move assignment operator.

Move assignment operator implementation.

4.1.3.9 readStudent()

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

Reads student data from an input stream.

Read student data from an input stream.

Parameters

is	The input stream to read from.

Returns

istream& The input stream after reading the student data.

4.1.3.10 setEgzaminoRez()

```
void Studentas::setEgzaminoRez (
             double egzaminorez)  [inline]
```

Set the exam grade of the student.

4.1.3.11 setPavarde()

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

Set the last name of the student.

4.1.3.12 setVardas()

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

Set the first name of the student.

4.1.4 Member Data Documentation

4.1.4.1 galutinisbalasmediana_

```
double Studentas::galutinisbalasmediana_
```

Final grade calculated using median method.

4.1.4.2 galutinisbalasvidurkis_

```
double Studentas::galutinisbalasvidurkis_
```

Final grade calculated using average method.

4.1.4.3 mediana_

```
double Studentas::mediana_
```

Median final grade.

4.1.4.4 namudarburezsuma_

```
double Studentas::namudarburezsuma_
```

Sum of homework grades.

4.1.4.5 vidurkis_

```
double Studentas::vidurkis_
```

Average final grade.

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

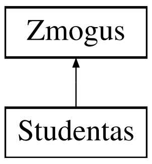
- src/Studentas.h
- src/Studentas.cpp

4.2 Zmogus Class Reference

Abstract class representing a person.

```
#include <Studentas.h>
```

Inheritance diagram for Zmogus:



Public Member Functions

- virtual ∼Zmogus ()=default
- virtual std::string getVardas () const =0

  Get the first name of the person.
- virtual std::string getPavarde () const =0

  Get the last name of the person.
- virtual double getEgzaminoRez () const =0

  Get the exam grade of the person.
- virtual std::vector< double > & getNamudarbuRez ()=0

  Get the grades of homework for the person.

4.2.1 Detailed Description

Abstract class representing a person.

This class defines the basic interface for a person, providing methods to retrieve information such as name and exam grades.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 ∼Zmogus()

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

4.2.3 Member Function Documentation

4.2.3.1 getEgzaminoRez()

```
virtual double Zmogus::getEgzaminoRez () const  [pure virtual]
```

Get the exam grade of the person.

Returns

The exam grade of the person.

Implemented in Studentas.

4.2.3.2 getNamudarbuRez()

```
virtual std::vector< double > & Zmogus::getNamudarbuRez ()  [pure virtual]
```

Get the grades of homework for the person.

Returns

A reference to the vector containing homework grades of the person.

Implemented in Studentas.

4.2.3.3 getPavarde()

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

Get the last name of the person.

Returns

The last name of the person.

Implemented in Studentas.

4.2.3.4 getVardas()

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

Get the first name of the person.

Returns

The first name of the person.

Implemented in Studentas.

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

- src/Studentas.h

Chapter 5

File Documentation

5.1 src/ConsoleApplication1.cpp File Reference

```
#include "MokiniuProcessing.h"
#include "Skaiciavimaidarbai.h"
```

Functions

- int main ()

  Main function that serves as the entry point of the program.

5.1.1 Function Documentation

5.1.1.1 main()

```
int main ()
```

Main function that serves as the entry point of the program.

The main function initializes the necessary components, handles user input, and directs the program flow based on the user's choices. It provides options for entering student data manually, generating random data, reading from files, and testing different strategies.

Returns

int Returns 0 on successful execution.

5.2 src/MokiniuProcessing.cpp File Reference

```
#include "MokiniuProcessing.h"
```

Functions

- string GeneruotiVardus ()

    Generates a random Lithuanian first name from a predefined list.
- string GeneruotiPavardes ()

    Generates a random Lithuanian last name from a predefined list.
- void PatikrintiTeigiamajiSkaiciu (double skaicius)

    Checks if a given number is within a valid range (0 to 10).
- bool ContainsNumbers (const string &str)

    Checks if a string contains any numeric digits.

5.2.1 Function Documentation

5.2.1.1 ContainsNumbers()

```
bool ContainsNumbers (
            const string & str)
```

Checks if a string contains any numeric digits.

This function returns true if the provided string contains at least one numeric digit, and false otherwise.

Parameters

str	The string to check.

Returns

bool True if the string contains numeric digits, false otherwise.

5.2.1.2 GeneruotiPavardes()

```
string GeneruotiPavardes ()
```

Generates a random Lithuanian last name from a predefined list.

This function selects a random last name from a vector of common Lithuanian surnames and returns it as a string.

Returns

string A randomly selected Lithuanian last name.

5.2.1.3 GeneruotiVardus()

```
string GeneruotiVardus ()
```

Generates a random Lithuanian first name from a predefined list.

This function selects a random first name from a vector of common Lithuanian names and returns it as a string.

Returns

string A randomly selected Lithuanian first name.

5.2.1.4 PatikrintiTeigiamajiSkaiciu()

```
void PatikrintiTeigiamajiSkaiciu (
             double skaicius)
```

Checks if a given number is within a valid range (0 to 10).

This function throws an invalid_argument exception if the provided number is not within the range of 0 to 10.

Parameters

skaicius	The number to check.

Exceptions

invalid_argument	If the number is not within the range 0 to 10.

5.3 src/MokiniuProcessing.h File Reference

```
#include <string>
#include <vector>
#include <stdexcept>
#include <iostream>
#include <locale>
#include <numeric>
#include <fstream>
#include <sstream>
#include <cctype>
#include <algorithm>
#include <chrono>
#include <iomanip>
#include "Studentas.h"
```

Functions

- string GeneruotiVardus ()

  Generates a random Lithuanian first name from a predefined list.
- string GeneruotiPavardes ()

  Generates a random Lithuanian last name from a predefined list.
- void PatikrintiTeigiamajiSkaiciu (double skaicius)

  Checks if a given number is within a valid range (0 to 10).
- bool ContainsNumbers (const string &str)

  Checks if a string contains any numeric digits.

5.3.1 Function Documentation

5.3.1.1 ContainsNumbers()

```
bool ContainsNumbers (
            const string & str)
```

Checks if a string contains any numeric digits.

This function returns true if the provided string contains at least one numeric digit, and false otherwise.

Parameters

str	The string to check.

Returns

bool True if the string contains numeric digits, false otherwise.

5.3.1.2 GeneruotiPavardes()

```
string GeneruotiPavardes ()
```

Generates a random Lithuanian last name from a predefined list.

This function selects a random last name from a vector of common Lithuanian surnames and returns it as a string.

Returns

string A randomly selected Lithuanian last name.

5.3.1.3 GeneruotiVardus()

```
string GeneruotiVardus ()
```

Generates a random Lithuanian first name from a predefined list.

This function selects a random first name from a vector of common Lithuanian names and returns it as a string.

Returns

string A randomly selected Lithuanian first name.

5.3.1.4 PatikrintiTeigiamajiSkaiciu()

```
void PatikrintiTeigiamajiSkaiciu (
             double skaicius)
```

Checks if a given number is within a valid range (0 to 10).

This function throws an invalid_argument exception if the provided number is not within the range of 0 to 10.

Parameters

skaicius	The number to check.

Exceptions

invalid_argument	If the number is not within the range 0 to 10.

5.4 MokiniuProcessing.h

Go to the documentation of this file.

```
#pragma once
#include <string>
#include <vector>
#include <stdexcept>
#include <iostream>
#include <locale>
#include <numeric>
#include <fstream>
#include <sstream>
#include <cctype>
#include <algorithm>
#include <chrono>
#include <iomanip>
#include "Studentas.h"

using namespace std;

string GeneruotiVardus();

string GeneruotiPavardes();

void PatikrintiTeigiamajiSkaiciu(double skaicius);

bool ContainsNumbers(const string& str);
```

5.5 src/Skaiciavimaidarbai.cpp File Reference

```
#include "Skaiciavimaidarbai.h"
#include "MokiniuProcessing.h"
#include "Studentas.h"
```

Functions

- void NetinkamaIvestis ()

  Prints a message indicating invalid input and terminates the program.
- void NeraFailo ()

  Prints a message indicating that a file was not found and terminates the program.
- double Mediana (vector< double > &vec)

  Calculates the median of a vector of doubles.
- double GenerateRandomGrade ()

  Generates a random grade between 0 and 10.
- void GeneruotiFaila (const string &pavadinimas, int studentuskaicius)

  Generates a file with random student data.
- bool compareByGalutinisVid (const Studentas &a, const Studentas &b)

  Comparator function to compare students by their final average grade.
- bool compareByGalutinisMed (const Studentas &a, const Studentas &b)

  Comparator function to compare students by their final median grade.

5.5.1 Function Documentation

5.5.1.1 compareByGalutinisMed()

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

Comparator function to compare students by their final median grade.

This function compares two students based on their final median grade and returns true if the first student's grade is less than the second student's grade.

Parameters

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

Returns

bool True if the first student's final median grade is less than the second student's grade.

5.5.1.2 compareByGalutinisVid()

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

Comparator function to compare students by their final average grade.

This function compares two students based on their final average grade and returns true if the first student's grade is less than the second student's grade.

Parameters

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

Returns

bool True if the first student's final average grade is less than the second student's grade.

5.5.1.3 GenerateRandomGrade()

```
double GenerateRandomGrade ()
```

Generates a random grade between 0 and 10.

This function returns a random grade in the range of 0 to 10.

Returns

double A randomly generated grade between 0 and 10.

5.5.1.4 GeneruotiFaila()

```
void GeneruotiFaila (
             const string & pavadinimas,
             int studentuskaicius)
```

Generates a file with random student data.

This function creates a file with the specified name and populates it with a specified number of students. Each student will have a generated name, surname, and a set of random grades.

Parameters

pavadinimas	The name of the file to generate.
studentuskaicius	The number of students to generate data for.

5.5.1.5 Mediana()

```
double Mediana (
             vector< double > & vec)
```

Calculates the median of a vector of doubles.

This function sorts the provided vector and calculates the median value. If the vector size is even, it returns the average of the two middle elements. If the vector size is odd, it returns the middle element.

Parameters

vec	The vector of doubles to calculate the median for.

Returns

double The median value of the vector.

5.5.1.6 NeraFailo()

```
void NeraFailo ()
```

Prints a message indicating that a file was not found and terminates the program.

This function outputs a message to the console indicating that the specified file was not found and then terminates the program.

5.5.1.7 NetinkamaIvestis()

```
void NetinkamaIvestis ()
```

Prints a message indicating invalid input and terminates the program.

This function outputs a message to the console indicating that the input was invalid and then terminates the program.

5.6 src/Skaiciavimaidarbai.h File Reference

```
#include "MokiniuProcessing.h"
```

Functions

- void NetinkamaIvestis ()

  Prints a message indicating invalid input and terminates the program.
- void NeraFailo ()

  Prints a message indicating that a file was not found and terminates the program.
- double Mediana (std::vector< double > &vec)

  Calculates the median of a vector of doubles.
- double GenerateRandomGrade ()

  Generates a random grade between 0 and 10.
- void GeneruotiFaila (const string &pavadinimas, int studentuskaicius)

  Generates a file with random student data.
- bool compareByGalutinisVid (const Studentas &a, const Studentas &b)

  Comparator function to compare students by their final average grade.
- bool compareByGalutinisMed (const Studentas &a, const Studentas &b)

  Comparator function to compare students by their final median grade.

5.6.1 Function Documentation

5.6.1.1 compareByGalutinisMed()

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

Comparator function to compare students by their final median grade.

This function compares two students based on their final median grade and returns true if the first student's grade is less than the second student's grade.

Parameters

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

Returns

bool True if the first student's final median grade is less than the second student's grade.

5.6.1.2 compareByGalutinisVid()

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

Comparator function to compare students by their final average grade.

This function compares two students based on their final average grade and returns true if the first student's grade is less than the second student's grade.

Parameters

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

Returns

bool True if the first student's final average grade is less than the second student's grade.

5.6.1.3 GenerateRandomGrade()

```
double GenerateRandomGrade ()
```

Generates a random grade between 0 and 10.

This function returns a random grade in the range of 0 to 10.

Returns

double A randomly generated grade between 0 and 10.

5.6.1.4 GeneruotiFaila()

```
void GeneruotiFaila (
             const string & pavadinimas,
             int studentuskaicius)
```

Generates a file with random student data.

This function creates a file with the specified name and populates it with a specified number of students. Each student will have a generated name, surname, and a set of random grades.

Parameters

pavadinimas	The name of the file to generate.
studentuskaicius	The number of students to generate data for.

5.6.1.5 Mediana()

```
double Mediana (
             std::vector< double > & vec)
```

Calculates the median of a vector of doubles.

This function sorts the provided vector and calculates the median value. If the vector size is even, it returns the average of the two middle elements. If the vector size is odd, it returns the middle element.

Parameters

vec	The vector of doubles to calculate the median for.

Returns

double The median value of the vector.

5.6.1.6 NeraFailo()

```
void NeraFailo ()
```

Prints a message indicating that a file was not found and terminates the program.

This function outputs a message to the console indicating that the specified file was not found and then terminates the program.

5.6.1.7 NetinkamaIvestis()

```
void NetinkamaIvestis ()
```

Prints a message indicating invalid input and terminates the program.

This function outputs a message to the console indicating that the input was invalid and then terminates the program.

5.7 Skaiciavimaidarbai.h

Go to the documentation of this file.

```
#pragma once
#include "MokiniuProcessing.h"

void NetinkamaIvestis();

void NeraFailo();

double Mediana(std::vector<double>& vec);

double GenerateRandomGrade();

void GeneruotiFaila(const string& pavadinimas, int studentuskaicius);

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

bool compareByGalutinisMed(const Studentas& a, const Studentas& b);
```

5.8 src/Studentas.cpp File Reference

```
#include "Studentas.h"
#include "Skaiciavimaidarbai.h"
#include <fstream>
#include <chrono>
#include <algorithm>
```

Functions

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

  Compares two students by their first name.
- bool pavardeslyginimas (const Studentas &a, const Studentas &b)

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

  Compares two students by their final average grade.
- bool medianoslyginimas (const Studentas &a, const Studentas &b)

  Compares two students by their final median grade.
- void PrintStudents (const vector< Studentas > &studentai)

  Prints a list of students to the console.
- void WriteNormalStudents (std::vector< Studentas > &normalus)

  Writes normal students to a file.
- void WriteWeirdStudents (std::vector< Studentas > &nenormalus)

  Writes weird students to a file.
- void readAndProcessData (const std::string &filename, std::vector< Studentas > &studentai, int &namudarbai, int studentuskaicius)

  Reads and processes student data from a file.
- void sortStudents (vector< Studentas > &studentai, int sortpasirinkimas)

  Sorts students using the STL sort function.
- void partitionStudents1 (const vector< Studentas > &studentai, vector< Studentas > &normalus, vector< Studentas > &nenormalus)

  Partitions students into normal and not normal students using method 1.
- void partitionStudents2 (vector< Studentas > &studentai, vector< Studentas > &nenormalus)

  Partitions students into normal and not normal students using method 2.
- void partitionStudents3 (vector< Studentas > &studentai, vector< Studentas > &normalus, vector< Studentas > &nenormalus)

  Partitions students into normal and not normal students using method 3.
- std::ostream & operator<< (std::ostream &os, Studentas &studentas)

  Output operator implementation.
- std::istream & operator>> (std::istream &is, Studentas &studentas)

  Input operator implementation.
- void testConstructors ()

  Test constructors.

5.8.1 Function Documentation

5.8.1.1 medianoslyginimas()

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

Compares two students by their final median grade.

Parameters

a	The first student.
b	The second student.

Returns

bool True if the first student's final median grade is less than the second student's final median grade.

5.8.1.2 operator<<()

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

Output operator implementation.

Output operator implementation.

5.8.1.3 operator>>()

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

Input operator implementation.

Input operator implementation.

5.8.1.4 partitionStudents1()

```
void partitionStudents1 (
            const vector< Studentas > & studentai,
            vector< Studentas > & normalus,
            vector< Studentas > & nenormalus)
```

Partitions students into normal and not normal students using method 1.

Parameters

studentai	The vector to store the read students.
normalus	The vector to store normal students.
nenormalus	The vector to store not normal students.

5.8.1.5 partitionStudents2()

```
void partitionStudents2 (
            vector< Studentas > & studentai,
            vector< Studentas > & nenormalus)
```

Partitions students into normal and not normal students using method 2.

Parameters

studentai	The vector to store the read students.
nenormalus	The vector to store not normal students.

5.8.1.6 partitionStudents3()

```
void partitionStudents3 (
            vector< Studentas > & studentai,
            vector< Studentas > & normalus,
            vector< Studentas > & nenormalus)
```

Partitions students into normal and not normal students using method 3.

Parameters

studentai	The vector to store the read students.
normalus	The vector to store normal students.
nenormalus	The vector to store not normal students.

5.8.1.7 pavardeslyginimas()

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

Compares two students by their last name.

Parameters

a	The first student.
b	The second student.

Returns

bool True if the first student's last name is less than the second student's last name.

5.8.1.8 PrintStudents()

```
void PrintStudents (
            const vector< Studentas > & studentai)
```

Prints a list of students to the console.

Parameters

studentai	The vector of students to print.

5.8.1.9 readAndProcessData()

```
void readAndProcessData (
            const std::string & filename,
            std::vector< Studentas > & studentai,
            int & namudarbai,
            int studentuskaicius)
```

Reads and processes student data from a file.

Parameters

filename	The name of the file to read from.
studentai	The vector to store the read students.
namudarbai	The number of homework grades.
studentuskaicius	The number of students.

5.8.1.10 sortStudents()

```
void sortStudents (
            vector< Studentas > & studentai,
            int sortpasirinkimas)
```

Sorts students using the STL sort function.

Parameters

studentai	The vector to store the read students.
sortpasirinkimas	The sorting option to use.

5.8.1.11 testConstructors()

```
void testConstructors ()
```

Test constructors.

Test constructors.

5.8.1.12 vardolyginimas()

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

Compares two students by their first name.

Parameters

a	The first student.
b	The second student.

Returns

bool True if the first student's name is less than the second student's name.

5.8.1.13 vidurkiolyginimas()

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

Compares two students by their final average grade.

Parameters

a	The first student.
b	The second student.

Returns

bool True if the first student's final average grade is less than the second student's final average grade.

5.8.1.14 WriteNormalStudents()

```
void WriteNormalStudents (
             std::vector< Studentas > & normalus)
```

Writes normal students to a file.

Parameters

normalus	The vector of normal students.

5.8.1.15 WriteWeirdStudents()

```
void WriteWeirdStudents (
             std::vector< Studentas > & nenormalus)
```

Writes weird students to a file.

Parameters

nenormalus	The vector of weird students.

5.9 src/Studentas.h File Reference

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

Classes

- class Zmogus

    Abstract class representing a person.
- class Studentas

    Class representing a student.

Functions

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

    Compares two students by their first name.
- bool pavardeslyginimas (const Studentas &a, const Studentas &b)

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

    Compares two students by their final average grade.
- bool medianoslyginimas (const Studentas &a, const Studentas &b)

    Compares two students by their final median grade.
- void PrintStudents (const std::vector< Studentas > &studentai)

- void readAndProcessData (const std::string &filename, std::vector< Studentas > &studentai, int &namudarbai, int studentuskaicius)

  Reads and processes student data from a file.
- void sortStudents (std::vector< Studentas > &studentai, int sortpasirinkimas)
- void partitionStudents1 (const std::vector< Studentas > &studentai, std::vector< Studentas > &normalus, std::vector< Studentas > &nenormalus)
- void partitionStudents2 (std::vector< Studentas > &studentai, std::vector< Studentas > &nenormalus)
- void partitionStudents3 (std::vector< Studentas > &studentai, std::vector< Studentas > &normalus, std↩ ::vector< Studentas > &nenormalus)
- void WriteWeirdStudents (std::vector< Studentas > &nenormalus)

  Writes weird students to a file.
- void WriteNormalStudents (std::vector< Studentas > &normalus)

  Writes normal students to a file.
- void testConstructors ()

  Test constructors.

5.9.1 Function Documentation

5.9.1.1 medianoslyginimas()

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

Compares two students by their final median grade.

Compare students by median final grade.

Parameters

a	The first student.
b	The second student.

Returns

bool True if the first student's final median grade is less than the second student's final median grade.

5.9.1.2 partitionStudents1()

```
void partitionStudents1 (
            const std::vector< Studentas > & studentai,
            std::vector< Studentas > & normalus,
            std::vector< Studentas > & nenormalus)
```

Partition students into normal and weird categories (method 1).

5.9.1.3 partitionStudents2()

```
void partitionStudents2 (
            std::vector< Studentas > & studentai,
            std::vector< Studentas > & nenormalus)
```

Partition students into normal and weird categories (method 2).

5.9.1.4 partitionStudents3()

```
void partitionStudents3 (
            std::vector< Studentas > & studentai,
            std::vector< Studentas > & normalus,
            std::vector< Studentas > & nenormalus)
```

Partition students into normal and weird categories (method 3).

5.9.1.5 pavardeslyginimas()

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

Compares two students by their last name.

Compare students by last name.

Parameters

a	The first student.
b	The second student.

Returns

bool True if the first student's last name is less than the second student's last name.

5.9.1.6 PrintStudents()

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

Print a list of students.

5.9.1.7 readAndProcessData()

```
void readAndProcessData (
            const std::string & filename,
            std::vector< Studentas > & studentai,
            int & namudarbai,
            int studentuskaicius)
```

Reads and processes student data from a file.

Read and process student data from a file.

Parameters

filename	The name of the file to read from.
studentai	The vector to store the read students.
namudarbai	The number of homework grades.
studentuskaicius	The number of students.

5.9.1.8 sortStudents()

```
void sortStudents (
             std::vector< Studentas > & studentai,
             int sortpasirinkimas)
```

Sort students based on specified criteria.

5.9.1.9 testConstructors()

```
void testConstructors ()
```

Test constructors.

Test constructors of the Studentas class.

Test constructors.

5.9.1.10 vardolyginimas()

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

Compares two students by their first name.

Compare students by first name.

Parameters

a	The first student.
b	The second student.

Returns

bool True if the first student's name is less than the second student's name.

5.9.1.11 vidurkiolyginimas()

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

Compares two students by their final average grade.

Compare students by average final grade.

Parameters

a	The first student.
b	The second student.

Returns

bool True if the first student's final average grade is less than the second student's final average grade.

5.9.1.12 WriteNormalStudents()

```
void WriteNormalStudents (
             std::vector< Studentas > & normalus)
```

Writes normal students to a file.

Write normal students to a file.

Parameters

normalus	The vector of normal students.

5.9.1.13 WriteWeirdStudents()

```
void WriteWeirdStudents (
             std::vector< Studentas > & nenormalus)
```

Writes weird students to a file.

Write weird students to a file.

Parameters

nenormalus	The vector of weird students.

5.10 Studentas.h

Go to the documentation of this file.

```
#pragma once

#include <string>
#include <vector>

class Zmogus {
public:
    virtual ~Zmogus() = default;
    // Getters
    virtual std::string getVardas() const = 0;
    virtual std::string getPavarde() const = 0;
    virtual double getEgzaminoRez() const = 0;
    virtual std::vector<double>& getNamudarbuRez() = 0;
};
class Studentas : public Zmogus {
private:
    std::string vardas_;
    std::string pavarde_;
    std::vector<double> namudarburez_;
    double egzaminorez_;

public:
    double namudarburezsuma_;
    double vidurkis_;
    double galutinisbalasvidurkis_;
    double mediana_;
    double galutinisbalasmediana_;
    // Constructors
    Studentas();
    Studentas(std::istream& is);
    // Getters
    std::string getVardas() const { return vardas_; }
    std::string getPavarde() const { return pavarde_; }
    double getEgzaminoRez() const { return egzaminorez_; }
    std::vector<double>& getNamudarbuRez() { return namudarburez_; }
    // Setters
    void setVardas(const std::string& vardas) { vardas_ = vardas; }
    void setPavarde(const std::string& pavarde) { pavarde_ = pavarde; }
    void setEgzaminoRez(double egzaminorez) { egzaminorez_ = egzaminorez; }
    void addGrade(double grade) { namudarburez_.push_back(grade); }
    // Other member functions
    std::istream& readStudent(std::istream&);
    void calculateFinalGrades();
    //Destructor
    ~Studentas();
    Studentas(const Studentas& other);
    Studentas& operator=(const Studentas& other);
    Studentas(Studentas&& other) noexcept;
    Studentas& operator=(Studentas&& other) noexcept;
};
// Negalima kurti zmogus objekto -
// Zmogus a;

// Comparison functions
bool vardolyginimas(const Studentas& a, const Studentas& b);
bool pavardeslyginimas(const Studentas& a, const Studentas& b);
bool vidurkiolyginimas(const Studentas& a, const Studentas& b);
bool medianoslyginimas(const Studentas& a, const Studentas& b);
// Utility functions
void PrintStudents(const std::vector<Studentas>& studentai);
void readAndProcessData(const std::string& filename, std::vector<Studentas>& studentai, int&
namudarbai, int studentuskaicius);
void sortStudents(std::vector<Studentas>& studentai, int sortpasirinkimas);
void partitionStudents1(const std::vector<Studentas>& studentai, std::vector<Studentas>& normalus,
std::vector<Studentas>& nenormalus);
void partitionStudents2(std::vector<Studentas>& studentai, std::vector<Studentas>& nenormalus);
void partitionStudents3(std::vector<Studentas>& studentai, std::vector<Studentas>& normalus,
std::vector<Studentas>& nenormalus);
void WriteWeirdStudents(std::vector<Studentas>& nenormalus);
void WriteNormalStudents(std::vector<Studentas>& normalus);
void testConstructors();
```

Index