

My Project

Generated by Doxygen 1.11.0

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Class Documentation	7
4.1 Studentas Class Reference	7
4.1.1 Detailed Description	8
4.1.2 Constructor & Destructor Documentation	8
4.1.2.1 Studentas() [1/4]	8
4.1.2.2 Studentas() [2/4]	8
4.1.2.3 ~Studentas()	9
4.1.2.4 Studentas() [3/4]	9
4.1.2.5 Studentas() [4/4]	9
4.1.3 Member Function Documentation	9
4.1.3.1 addGrade()	9
4.1.3.2 calculateFinalGrades()	10
4.1.3.3 getEgzaminoRez()	10
4.1.3.4 getNamudarbuRez()	10
4.1.3.5 getPavarde()	10
4.1.3.6 getVardas()	10
4.1.3.7 operator=() [1/2]	10
4.1.3.8 operator=() [2/2]	11
4.1.3.9 readStudent()	11
4.1.3.10 setEgzaminoRez()	11
4.1.3.11 setPavarde()	11
4.1.3.12 setVardas()	11
4.1.4 Member Data Documentation	12
4.1.4.1 galutinisbalasmediana_	12
4.1.4.2 galutinisbalasvidurkis_	12
4.1.4.3 mediana_	12
4.1.4.4 namudarburezsuma_	12
4.1.4.5 vidurkis_	12
4.2 Zmogus Class Reference	12
4.2.1 Detailed Description	13
4.2.2 Constructor & Destructor Documentation	13
4.2.2.1 ~Zmogus()	13
4.2.3 Member Function Documentation	13
4.2.3.1 getEgzaminoRez()	13

4.2.3.2 getNamudarbuRez()	13
4.2.3.3 getPavarde()	14
4.2.3.4 getVardas()	14
5 File Documentation	15
5.1 src/ConsoleApplication1.cpp File Reference	15
5.1.1 Function Documentation	15
5.1.1.1 main()	15
5.2 src/MokiniuProcessing.cpp File Reference	15
5.2.1 Function Documentation	16
5.2.1.1 ContainsNumbers()	16
5.2.1.2 GeneruotiPavardes()	16
5.2.1.3 GeneruotiVardus()	16
5.2.1.4 PatikrintiTeigiamajiSkaiciu()	17
5.3 src/MokiniuProcessing.h File Reference	18
5.3.1 Function Documentation	18
5.3.1.1 ContainsNumbers()	18
5.3.1.2 GeneruotiPavardes()	19
5.3.1.3 GeneruotiVardus()	19
5.3.1.4 PatikrintiTeigiamajiSkaiciu()	19
5.4 MokiniuProcessing.h	20
5.5 src/Skaiciavimaidarbai.cpp File Reference	20
5.5.1 Function Documentation	21
5.5.1.1 compareByGalutinisMed()	21
5.5.1.2 compareByGalutinisVid()	21
5.5.1.3 GenerateRandomGrade()	21
5.5.1.4 GeneruotiFaila()	22
5.5.1.5 Mediana()	22
5.5.1.6 NeraFailo()	22
5.5.1.7 Netinkamalvestis()	22
5.6 src/Skaiciavimaidarbai.h File Reference	23
5.6.1 Function Documentation	23
5.6.1.1 compareByGalutinisMed()	23
5.6.1.2 compareByGalutinisVid()	23
5.6.1.3 GenerateRandomGrade()	24
5.6.1.4 GeneruotiFaila()	24
5.6.1.5 Mediana()	24
5.6.1.6 NeraFailo()	25
5.6.1.7 Netinkamalvestis()	25
5.7 Skaiciavimaidarbai.h	25
5.8 src/Studentas.cpp File Reference	25
5.8.1 Function Documentation	26

5.8.1.1 medianoslyginimas()	26
5.8.1.2 operator<<()	27
5.8.1.3 operator>>()	27
5.8.1.4 partitionStudents1()	27
5.8.1.5 partitionStudents2()	27
5.8.1.6 partitionStudents3()	27
5.8.1.7 pavardeslyginimas()	28
5.8.1.8 PrintStudents()	28
5.8.1.9 readAndProcessData()	28
5.8.1.10 sortStudents()	29
5.8.1.11 testConstructors()	29
5.8.1.12 vardolyginimas()	29
5.8.1.13 vidurkiolyginimas()	29
5.8.1.14 WriteNormalStudents()	30
5.8.1.15 WriteWeirdStudents()	30
5.9 src/Studentas.h File Reference	30
5.9.1 Function Documentation	31
5.9.1.1 medianoslyginimas()	31
5.9.1.2 partitionStudents1()	31
5.9.1.3 partitionStudents2()	31
5.9.1.4 partitionStudents3()	32
5.9.1.5 pavardeslyginimas()	32
5.9.1.6 PrintStudents()	32
5.9.1.7 readAndProcessData()	32
5.9.1.8 sortStudents()	33
5.9.1.9 testConstructors()	33
5.9.1.10 vardolyginimas()	33
5.9.1.11 vidurkiolyginimas()	33
5.9.1.12 WriteNormalStudents()	34
5.9.1.13 WriteWeirdStudents()	34
5.10 Studentas.h	34
Index	37

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Zmogus	12
Studentas	7

Chapter 2

Class Index

2.1 Class List

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

Studentas	Class representing a student	7
Zmogus	Abstract class representing a person	12

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

src/ ConsoleApplication1.cpp	15
src/ MokiniuProcessing.cpp	15
src/ MokiniuProcessing.h	18
src/ Skaiciavimaidarbai.cpp	20
src/ Skaiciavimaidarbai.h	23
src/ Studentas.cpp	25
src/ Studentas.h	30

Chapter 4

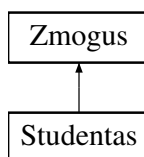
Class Documentation

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 implementation
constructor 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

<i>is</i>	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

<i>is</i>	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:

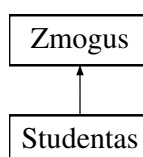
- [src/Studentas.h](#)
- [src/Studentas.cpp](#)

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

<i>str</i>	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

<i>skaicius</i>	The number to check.
-----------------	----------------------

Exceptions

<i>invalid_argument</i>	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

<i>str</i>	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

<i>skaicius</i>	The number to check.
-----------------	----------------------

Exceptions

<code>invalid_argument</code>	If the number is not within the range 0 to 10.
-------------------------------	--

5.4 MokiniuProcessing.h

[Go to the documentation of this file.](#)

```

00001 #pragma once
00002 #include <string>
00003 #include <vector>
00004 #include <stdexcept>
00005 #include <iostream>
00006 #include <locale>
00007 #include <numeric>
00008 #include <fstream>
00009 #include <sstream>
00010 #include <cctype>
00011 #include <algorithm>
00012 #include <chrono>
00013 #include <iomanip>
00014 #include "Studentas.h"
00015
00016 using namespace std;
00017
00026 string GeneruotiVardus();
00027
00036 string GeneruotiPavardes();
00037
00047 void PatikrintiTeigiamajiSkaiciu(double skaicius);
00048
00058 bool ContainsNumbers(const string& str);

```

5.5 src/Skaiciavimaidarbai.cpp File Reference

```

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

```

Functions

- void [Netinkamalvestis](#) ()
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

<i>a</i>	The first student to compare.
<i>b</i>	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

<i>a</i>	The first student to compare.
<i>b</i>	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

<i>pavadinimas</i>	The name of the file to generate.
<i>studentuskaicius</i>	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

<i>vec</i>	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 Netinkamalvestis()

```
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 [Netinkamalvestis](#) ()
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

<i>a</i>	The first student to compare.
<i>b</i>	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

<i>a</i>	The first student to compare.
<i>b</i>	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

<i>pavadinimas</i>	The name of the file to generate.
<i>studentuskaicius</i>	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 Netinkamalvestis()

```
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.](#)

```
00001 #pragma once
00002 #include "MokiniuProcessing.h"
00003
00010 void NetinkamaIvestis();
00011
00018 void NeraFailo();
00019
00030 double Mediana(std::vector<double>& vec);
00031
00039 double GenerateRandomGrade();
00040
00051 void GeneruotiFaila(const string& pavadinimas, int studentuskaicius);
00052
00063 bool compareByGalutinisVid(const Studentas& a, const Studentas& b);
00064
00075 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

<i>a</i>	The first student.
<i>b</i>	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

<i>studentai</i>	The vector to store the read students.
<i>normalus</i>	The vector to store normal students.
<i>nenormalus</i>	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

<i>studentai</i>	The vector to store the read students.
<i>nenormalus</i>	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

<i>studentai</i>	The vector to store the read students.
<i>normalus</i>	The vector to store normal students.
<i>nenormalus</i>	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

<i>a</i>	The first student.
<i>b</i>	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

<i>studentai</i>	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

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

5.8.1.10 sortStudents()

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

Sorts students using the STL sort function.

Parameters

<i>studentai</i>	The vector to store the read students.
<i>sortpasirinkimas</i>	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

<i>a</i>	The first student.
<i>b</i>	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

<i>a</i>	The first student.
<i>b</i>	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

<i>normalus</i>	The vector of normal students.
-----------------	--------------------------------

5.8.1.15 WriteWeirdStudents()

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

Writes weird students to a file.

Parameters

<i>nenormalus</i>	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

<i>a</i>	The first student.
<i>b</i>	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

<i>a</i>	The first student.
<i>b</i>	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

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

<i>a</i>	The first student.
<i>b</i>	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

<i>a</i>	The first student.
<i>b</i>	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

<i>normalus</i>	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

<i>nenormalus</i>	The vector of weird students.
-------------------	-------------------------------

5.10 Studentas.h

[Go to the documentation of this file.](#)

```
00001 #pragma once
00002
00003 #include <string>
00004 #include <vector>
00005
00012 class Zmogus {
00013 public:
00014     virtual ~Zmogus() = default;
00015     // Getters
00021     virtual std::string getVardas() const = 0;
00027     virtual std::string getPavarde() const = 0;
00033     virtual double getEgzaminoRez() const = 0;
00039     virtual std::vector<double> getNamudarburez() = 0;
00040 };
00047 class Studentas : public Zmogus {
00048 private:
00049     std::string vardas_;
00050     std::string pavarde_;
00051     std::vector<double> namudarburez_;
00052     double egzaminorez_;
00053
00054 public:
00055     double namudarburezsuma_;
00056     double vidurkis_;
00057     double galutinisbalasvidurkis_;
00058     double mediana_;
00059     double galutinisbalasmediana_;
00061     // Constructors
00062     Studentas();
00063     Studentas(std::istream& is);
00065     // Getters
00066     std::string getVardas() const { return vardas_; }
00067     std::string getPavarde() const { return pavarde_; }
00068     double getEgzaminoRez() const { return egzaminorez_; }
```



```
00069     std::vector<double>& getNamudarbuRez() { return namudarburez_; }
00071     // Setters
00072     void setVardas(const std::string& vardas) { vardas_ = vardas; }
00073     void setPavarde(const std::string& pavarde) { pavarde_ = pavarde; }
00074     void setEgzaminoRez(double egzaminorez) { egzaminorez_ = egzaminorez; }
00075     void addGrade(double grade) { namudarburez_.push_back(grade); }
00077     // Other member functions
00078     std::istream& readStudent(std::istream&);
00079     void calculateFinalGrades();
00081     //Destructor
00082     ~Studentas();
00083     Studentas(const Studentas& other);
00084     Studentas& operator=(const Studentas& other);
00085     Studentas(Studentas&& other) noexcept;
00086     Studentas& operator=(Studentas&& other) noexcept;
00088 };
00089 // Negalima kurti zmogus objekto -
00090 // Zmogus a;
00091
00092 // Comparison functions
00093 bool vardolyginimas(const Studentas& a, const Studentas& b);
00094 bool pavardeslyginimas(const Studentas& a, const Studentas& b);
00095 bool vidurkiolyginimas(const Studentas& a, const Studentas& b);
00096 bool medianoslyginimas(const Studentas& a, const Studentas& b);
00098 // Utility functions
00099 void PrintStudents(const std::vector<Studentas>& studentai);
00100 void readAndProcessData(const std::string& filename, std::vector<Studentas>& studentai, int&
    namudarbai, int studentuskaicius);
00101 void sortStudents(std::vector<Studentas>& studentai, int sortpasirinkimas);
00102 void partitionStudents1(const std::vector<Studentas>& studentai, std::vector<Studentas>& normalus,
    std::vector<Studentas>& nenormalus);
00103 void partitionStudents2(std::vector<Studentas>& studentai, std::vector<Studentas>& nenormalus);
00104 void partitionStudents3(std::vector<Studentas>& studentai, std::vector<Studentas>& normalus,
    std::vector<Studentas>& nenormalus);
00105 void WriteWeirdStudents(std::vector<Studentas>& nenormalus);
00106 void WriteNormalStudents(std::vector<Studentas>& normalus);
00107 void testConstructors();
```


Index

- ~Studentas
 - Studentas, [9](#)
- ~Zmogus
 - Zmogus, [13](#)
- addGrade
 - Studentas, [9](#)
- calculateFinalGrades
 - Studentas, [9](#)
- compareByGalutinisMed
 - Skaiciavimaidarbai.cpp, [21](#)
 - Skaiciavimaidarbai.h, [23](#)
- compareByGalutinisVid
 - Skaiciavimaidarbai.cpp, [21](#)
 - Skaiciavimaidarbai.h, [23](#)
- ConsoleApplication1.cpp
 - main, [15](#)
- ContainsNumbers
 - MokiniuProcessing.cpp, [16](#)
 - MokiniuProcessing.h, [18](#)
- galutinisbalasmediana_
 - Studentas, [12](#)
- galutinisbalasvidurkis_
 - Studentas, [12](#)
- GenerateRandomGrade
 - Skaiciavimaidarbai.cpp, [21](#)
 - Skaiciavimaidarbai.h, [24](#)
- GeneruotiFaila
 - Skaiciavimaidarbai.cpp, [21](#)
 - Skaiciavimaidarbai.h, [24](#)
- GeneruotiPavardes
 - MokiniuProcessing.cpp, [16](#)
 - MokiniuProcessing.h, [19](#)
- GeneruotiVardus
 - MokiniuProcessing.cpp, [16](#)
 - MokiniuProcessing.h, [19](#)
- getEgzaminoRez
 - Studentas, [10](#)
 - Zmogus, [13](#)
- getNamudarduRez
 - Studentas, [10](#)
 - Zmogus, [13](#)
- getPavarde
 - Studentas, [10](#)
 - Zmogus, [13](#)
- getVardas
 - Studentas, [10](#)
 - Zmogus, [14](#)
- main
 - ConsoleApplication1.cpp, [15](#)
- Mediana
 - Skaiciavimaidarbai.cpp, [22](#)
 - Skaiciavimaidarbai.h, [24](#)
- mediana_
 - Studentas, [12](#)
- medianoslyginimas
 - Studentas.cpp, [26](#)
 - Studentas.h, [31](#)
- MokiniuProcessing.cpp
 - ContainsNumbers, [16](#)
 - GeneruotiPavardes, [16](#)
 - GeneruotiVardus, [16](#)
 - PatikrintiTeigiamajiSkaiciu, [16](#)
- MokiniuProcessing.h
 - ContainsNumbers, [18](#)
 - GeneruotiPavardes, [19](#)
 - GeneruotiVardus, [19](#)
 - PatikrintiTeigiamajiSkaiciu, [19](#)
- namudarburezsuma_
 - Studentas, [12](#)
- NeraFailo
 - Skaiciavimaidarbai.cpp, [22](#)
 - Skaiciavimaidarbai.h, [25](#)
- Netinkamalvestis
 - Skaiciavimaidarbai.cpp, [22](#)
 - Skaiciavimaidarbai.h, [25](#)
- operator<<
 - Studentas.cpp, [26](#)
- operator>>
 - Studentas.cpp, [27](#)
- operator=
 - Studentas, [10](#)
- partitionStudents1
 - Studentas.cpp, [27](#)
 - Studentas.h, [31](#)
- partitionStudents2
 - Studentas.cpp, [27](#)
 - Studentas.h, [31](#)
- partitionStudents3
 - Studentas.cpp, [27](#)
 - Studentas.h, [31](#)
- PatikrintiTeigiamajiSkaiciu
 - MokiniuProcessing.cpp, [16](#)
 - MokiniuProcessing.h, [19](#)
- pavardeslyginimas

- Studentas.cpp, 28
- Studentas.h, 32
- PrintStudents
 - Studentas.cpp, 28
 - Studentas.h, 32
- readAndProcessData
 - Studentas.cpp, 28
 - Studentas.h, 32
- readStudent
 - Studentas, 11
- setEgzaminoRez
 - Studentas, 11
- setPavarde
 - Studentas, 11
- setVardas
 - Studentas, 11
- Skaiciavimaidarbai.cpp
 - compareByGalutinisMed, 21
 - compareByGalutinisVid, 21
 - GenerateRandomGrade, 21
 - GeneruotiFaila, 21
 - Mediana, 22
 - NeraFailo, 22
 - Netinkamalvestis, 22
- Skaiciavimaidarbai.h
 - compareByGalutinisMed, 23
 - compareByGalutinisVid, 23
 - GenerateRandomGrade, 24
 - GeneruotiFaila, 24
 - Mediana, 24
 - NeraFailo, 25
 - Netinkamalvestis, 25
- sortStudents
 - Studentas.cpp, 28
 - Studentas.h, 32
- src/ConsoleApplication1.cpp, 15
- src/MokiniuProcessing.cpp, 15
- src/MokiniuProcessing.h, 18, 20
- src/Skaiciavimaidarbai.cpp, 20
- src/Skaiciavimaidarbai.h, 23, 25
- src/Studentas.cpp, 25
- src/Studentas.h, 30, 34
- Studentas, 7
 - ~Studentas, 9
 - addGrade, 9
 - calculateFinalGrades, 9
 - galutinisbalasmediana_, 12
 - galutinisbalasvidurkis_, 12
 - getEgzaminoRez, 10
 - getNamudarbuRez, 10
 - getPavarde, 10
 - getVardas, 10
 - mediana_, 12
 - namudarburesuma_, 12
 - operator=, 10
 - readStudent, 11
 - setEgzaminoRez, 11
 - setPavarde, 11
 - setVardas, 11
 - Studentas, 8, 9
 - vidurkis_, 12
- Studentas.cpp
 - medianoslyginimas, 26
 - operator<<, 26
 - operator>>, 27
 - partitionStudents1, 27
 - partitionStudents2, 27
 - partitionStudents3, 27
 - pavardeslyginimas, 28
 - PrintStudents, 28
 - readAndProcessData, 28
 - sortStudents, 28
 - testConstructors, 29
 - vardolyginimas, 29
 - vidurkiolyginimas, 29
 - WriteNormalStudents, 29
 - WriteWeirdStudents, 30
- Studentas.h
 - medianoslyginimas, 31
 - partitionStudents1, 31
 - partitionStudents2, 31
 - partitionStudents3, 31
 - pavardeslyginimas, 32
 - PrintStudents, 32
 - readAndProcessData, 32
 - sortStudents, 32
 - testConstructors, 33
 - vardolyginimas, 33
 - vidurkiolyginimas, 33
 - WriteNormalStudents, 33
 - WriteWeirdStudents, 34
- testConstructors
 - Studentas.cpp, 29
 - Studentas.h, 33
- vardolyginimas
 - Studentas.cpp, 29
 - Studentas.h, 33
- vidurkiolyginimas
 - Studentas.cpp, 29
 - Studentas.h, 33
- vidurkis_
 - Studentas, 12
- WriteNormalStudents
 - Studentas.cpp, 29
 - Studentas.h, 33
- WriteWeirdStudents
 - Studentas.cpp, 30
 - Studentas.h, 34
- Zmogus, 12
 - ~Zmogus, 13
 - getEgzaminoRez, 13
 - getNamudarbuRez, 13

getPavarde, [13](#)
getVardas, [14](#)