Student Data Application

Generated by Doxygen 1.12.0

Chapter 1

Programos naudojimo instrukcija

1. Atsisiųsti v1.0 release paketą ir CMAKE 3.25 versiją;
2. Atsidaryti terminalą ir jame pakeisti dabartinę direktoriją į atsisiųsto (un-zip'into) kodo aplanką ("cd ./objektinis-pirma-obj-4" ar panašiai);
3. Terminale paleisti run.sh script'ą su komanda "./run.sh". *PASTABA*: jeigu nepavyksta, panaudoti komandą "chmod +x run.sh";
4. Programai pasileidus, skaityti ekrane pasirodžiusias instrukcijas ir jomis vadovautis.

1.0.1 Testavimo įranga:

- CPU: Apple M1
- RAM: 8 GB LPDDR4X-4266
- SSD: 256 GB ∼2700 MB/s

1.0.2 Programos versijų aprašas ir spartos analizė

1.0.2.1 Versija 0.1

- Generuoja studentų informaciją: namų darbai ir egzamino rezultatai
- Leidžia duomenis įvesti rankomis arba nuskaityti iš failo
- Informacija išvedama į failą sulygiuota ir išrikiuota pagal pavardę

Panaudotos bibliotekos: iostream, vector, sstream, random, algorithm, fstream, iomanip, stdexception.

1.0.2.2 Versija 0.2

- Generuoja studentų informaciją: namų darbai ir egzamino rezultatai
- Leidžia duomenis įvesti rankomis arba nuskaityti iš failo
- Informacija išvedama į failą sulygiuota ir išrikiuota pagal pavardę
- *Studentus suskirsto į "protingus" ir "kvailus" ir juos išveda į atskirus failus*

Panaudotos bibliotekos: iostream, vector, sstream, random, algorithm, fstream, iomanip, stdexception, chrono.

1.0.2.3 Versija 0.3

- Generuoja studentų informaciją: namų darbai ir egzamino rezultatai
- Leidžia duomenis įvesti rankomis arba nuskaityti iš failo
- Informacija išvedama į failą sulygiuota ir išrikiuota pagal pavardę
- Studentus suskirsto į "protingus" ir "kvailus" ir juos išveda į atskirus failus
- *Fiksuoja programos spartos laiką vector ir list konteineriams: įrašų generavimas, nuskaitymas, rikiavimas, studentų įrašų išskirtymas į "protingus" ir "kvailus"*

Panaudotos bibliotekos: iostream, vector, sstream, random, algorithm, fstream, iomanip, stdexception, chrono, list.

1.0.2.4 Versija 1.0

- Generuoja studentų informaciją: namų darbai ir egzamino rezultatai
- Leidžia duomenis įvesti rankomis arba nuskaityti iš failo
- Informacija išvedama į failą sulygiuota ir išrikiuota pagal pavardę
- Studentus suskirsto į "protingus" ir "kvailus" ir juos išveda į atskirus failus
- Fiksuoja programos spartos laiką vector ir list konteineriams: įrašų generavimas, nuskaitymas, rikiavimas, studentų įrašų išskirtymas į "protingus" ir "kvailus"
- *Studentų įrašų duomenis galima apdoroti pagal 3 skirtingas strategijas: (1) padalinti į naujus 2 konteinerius, (2) padalinti sukuriant tik 1 naują konteinerį, (3) taikant greičiausią būdą ir pritaikant efektyvius STL algoritmus*

Panaudotos bibliotekos: iostream, vector, sstream, random, algorithm, fstream, iomanip, stdexception, chrono, list, numeric.

1.0.2.5 Versija 1.1

1.0.3 Struct ir Class veikimo palyginimas

1.0.4 O1, O2 ir O3 optimizavimo flag'ų palyginimas

1.0.4.1 Versija 1.2

Įvedimo ir išvedimo member metodai class'ei Studentas buvo perdengti. Įvedimo metodas pilnai nuskaito duomenų eilutę sugeneruotame faile (vardas, pavardė ir visi gauti įvertinimai). Kai kuriais atvejais įvedimo metodas rankinio įvedimo metu nepanaudotas dėl skirtingų duomenų reikalavimų. Išvedimo metodas taip pat pritaikytas, nebent prašoma išvesti skirtigus duomenis (pvz. prašo vidurkio ir medianos, o metodas išveda tik vieną).

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

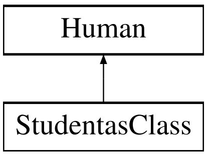
Class Documentation

5.1 Human Class Reference

Abstract father class for **StudentasClass** (p. 11).

`#include <studentas.h>`

Inheritance diagram for Human:



Public Member Functions

- virtual ~**Human** ()=0
- virtual void **setVardas** (const std::string &vardas)=0

  Set the First Name object.
- virtual void **setPavarde** (const std::string &pavarde)=0

  Set the Last Name object.
- virtual const std::string & **getVardas** () const =0

  Get the First Name object.
- virtual const std::string & **getPavarde** () const =0

  Get the Last Name object.

Protected Attributes

- std::string **vardas_**
- std::string **pavarde_**

5.1.1 Detailed Description

Abstract father class for **StudentasClass** (p. 11).

5.1.2 Constructor & Destructor Documentation

5.1.2.1 ∼Human()

```
Human::∼Human ()  [inline], [pure virtual]
```

5.1.3 Member Function Documentation

5.1.3.1 getPavarde()

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

Get the Last Name object.

Returns

const std::string&

Implemented in **StudentasClass** (p. 15).

5.1.3.2 getVardas()

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

Get the First Name object.

Returns

const std::string&

Implemented in **StudentasClass** (p. 16).

5.1.3.3 setPavarde()

```
virtual void Human::setPavarde (
             const std::string & pavarde)  [pure virtual]
```

Set the Last Name object.

Parameters

pavarde	

Implemented in **StudentasClass** (p. 18).

5.1.3.4 setVardas()

```
virtual void Human::setVardas (
             const std::string & vardas)  [pure virtual]
```

Set the First Name object.

Parameters

vardas	

Implemented in **StudentasClass** (p. 18).

5.1.4 Member Data Documentation

5.1.4.1 pavarde_

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

5.1.4.2 vardas_

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

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

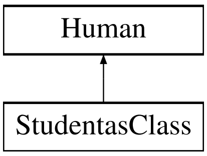
- include/ **studentas.h**

5.2 StudentasClass Class Reference

Student data processing.

```
#include <studentas.h>
```

Inheritance diagram for StudentasClass:



Public Member Functions

- **StudentasClass** ()

  Construct a new Studentas Class object.
- **StudentasClass** (const std::string &vardas, const std::string &pavarde)

  Construct a new Studentas Class object.
- ∼**StudentasClass** ()

  Destroy the Studentas Class object.
- **StudentasClass** (const **StudentasClass** &other)

  Construct a new Studentas Class object by copying data.
- **StudentasClass** & **operator=** (const **StudentasClass** &other)

  Construct a new Studentas Class object by assigning data.

- const std::string & **getVardas** () const override

    Get the First Name object.
- const std::string & **getPavarde** () const override

    Get the Last Name object.
- const std::vector< int > & **getTarpRez** () const

    Get the Intermediate Grades object.
- int **getEgzamRez** () const

    Get the Exam Grade object.
- double **getVidurkis** () const

    Get the Grade Average object.
- double **getMediana** () const

    Get the Grade Median object.
- double **getGalutinis** () const

    Get the Final Grade object.
- void **setVardas** (const std::string &vardas) override

    Set the First Name object.
- void **setPavarde** (const std::string &pavarde) override

    Set the Last Name object.
- void **setEgzamRez** (int rez)

    Set the Exam Grade object.
- void **setTarpRez** (const std::vector< int > &naujiTarpRez)

    Set the Intermediate Grades object.
- void **pridetiTarpRez** (int rez)

    Add a Single Intermediate Grade.
- void **generuotiBalus** (int kiekBalu=15)

    Generate random Intermediate Grades and Exam Grade.
- void **rastiGalutini** (bool naudotiVidurki=true)

    Calculate the Final Grade.
- void **isimtiGalutini** ()

    Remove the last Intermediate Grade and replace it with Exam Grade.
- void **clear** ()

    Remove all saved data.
- void **rastiIslaike** ()

    Update Passing Grade boolean islaike.
- bool **arIslaike** () const

    Recheck and return Passing Grade boolean.
- bool **compare** (const **StudentasClass** &b, int criteria=2)

    Compares Students either by First Name, Last Name or Final Grade.
- std::ostream & **printInfo** (std::ostream &os) const

Public Member Functions inherited from Human

- virtual ~**Human** ()=0

Friends

- std::istream & **operator>>** (std::istream &is, **StudentasClass** &s)
- std::ostream & **operator<<** (std::ostream &os, const **StudentasClass** &s)

Additional Inherited Members

Protected Attributes inherited from Human

- std::string **vardas_**
- std::string **pavarde_**

5.2.1 Detailed Description

Student data processing.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 StudentasClass() [1/3]

```
StudentasClass::StudentasClass ()  [inline]
```

Construct a new Studentas Class object.

5.2.2.2 StudentasClass() [2/3]

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

Construct a new Studentas Class object.

Parameters

vardas	
pavarde	

5.2.2.3 ∼StudentasClass()

```
StudentasClass::∼StudentasClass ()  [inline]
```

Destroy the Studentas Class object.

5.2.2.4 StudentasClass() [3/3]

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

Construct a new Studentas Class object by copying data.

Parameters

other	

5.2.3 Member Function Documentation

5.2.3.1 arIslaike()

```
bool StudentasClass::arIslaike () const  [inline]
```

Recheck and return Passing Grade boolean.

Returns

true

false

5.2.3.2 clear()

```
void StudentasClass::clear ()
```

Remove all saved data.

5.2.3.3 compare()

```
bool StudentasClass::compare (
             const  StudentasClass & b,
             int criteria = 2)
```

Compares Students either by First Name, Last Name or Final Grade.

Parameters

b	
criteria	

Returns

true

false

5.2.3.4 generuotiBalus()

```
void StudentasClass::generuotiBalus (
             int kiekBalu = 15)
```

Generate random Intermediate Grades and Exam Grade.

Parameters

kiekBalu	

5.2.3.5 getEgzamRez()

```
int StudentasClass::getEgzamRez () const  [inline]
```

Get the Exam Grade object.

Returns

int

5.2.3.6 getGalutinis()

```
double StudentasClass::getGalutinis () const  [inline]
```

Get the Final Grade object.

Returns

double

5.2.3.7 getMediana()

```
double StudentasClass::getMediana () const  [inline]
```

Get the Grade Median object.

Returns

double

5.2.3.8 getPavarde()

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

Get the Last Name object.

Returns

const std::string&

Implements **Human** (p. 10).

5.2.3.9 getTarpRez()

```
const std::vector< int > & StudentasClass::getTarpRez () const  [inline]
```

Get the Intermediate Grades object.

Returns

const std::vector<int>&

5.2.3.10 getVardas()

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

Get the First Name object.

Returns

const std::string&

Implements **Human** (p. 10).

5.2.3.11 getVidurkis()

```
double StudentasClass::getVidurkis () const  [inline]
```

Get the Grade Average object.

Returns

double

5.2.3.12 isimtiGalutini()

```
void StudentasClass::isimtiGalutini ()
```

Remove the last Intermediate Grade and replace it with Exam Grade.

5.2.3.13 operator=()

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

Construct a new Studentas Class object by assigning data.

Parameters

other	

Returns

StudentasClass (p. 11)&

5.2.3.14 pridetiTarpRez()

```
void StudentasClass::pridetiTarpRez (
             int rez)
```

Add a Single Intermediate Grade.

Parameters

rez	

5.2.3.15 printInfo()

```
std::ostream & StudentasClass::printInfo (
             std::ostream & os) const
```

5.2.3.16 rastiGalutini()

```
void StudentasClass::rastiGalutini (
             bool naudotiVidurki = true)
```

Calculate the Final Grade.

Parameters

naudotiVidurki	

5.2.3.17 rastiIslaike()

```
void StudentasClass::rastiIslaike () [inline]
```

Update Passing Grade boolean islaike.

5.2.3.18 setEgzamRez()

```
void StudentasClass::setEgzamRez (
             int rez) [inline]
```

Set the Exam Grade object.

Parameters

rez	

5.2.3.19 setPavarde()

```
void StudentasClass::setPavarde (
            const std::string & pavarde)  [inline], [override], [virtual]
```

Set the Last Name object.

Parameters

pavarde	

Implements **Human** (p. 10).

5.2.3.20 setTarpRez()

```
void StudentasClass::setTarpRez (
            const std::vector< int > & naujiTarpRez)
```

Set the Intermediate Grades object.

Parameters

naujiTarpRez	

5.2.3.21 setVardas()

```
void StudentasClass::setVardas (
            const std::string & vardas)  [inline], [override], [virtual]
```

Set the First Name object.

Parameters

vardas	

Implements **Human** (p. 10).

5.2.4 Friends And Related Symbol Documentation

5.2.4.1 operator<<

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

5.2.4.2 operator>>

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

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

- include/ **studentas.h**
- src/ **studentas.cpp**

5.3 Timer Class Reference

Tracks time for run-time measurements.

Public Member Functions

- **Timer** ()
- void **reset** ()
- double **elapsed** () const

5.3.1 Detailed Description

Tracks time for run-time measurements.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 Timer()

```
Timer::Timer ()  [inline]
```

5.3.3 Member Function Documentation

5.3.3.1 elapsed()

```
double Timer::elapsed () const  [inline]
```

5.3.3.2 reset()

```
void Timer::reset ()  [inline]
```

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

- src/ **failuGeneratorius.cpp**

Chapter 6

File Documentation

6.1 include/lib.h File Reference

```
#include <iostream>
#include <vector>
#include <sstream>
#include <random>
#include <algorithm>
#include <fstream>
#include <iomanip>
#include <stdexcept>
#include <chrono>
#include <list>
#include <numeric>
```

6.2 lib.h

Go to the documentation of this file.

```
#ifndef LIB_H_INCLUDED
#define LIB_H_INCLUDED

#include <iostream>
#include <vector>
#include <sstream> // getline išskirstymui
#include <random> // atsitiktiniai skaičiai
#include <algorithm> // sort
#include <fstream> // darbas su failais
#include <iomanip> // setw, setprecision, std::left, std::right
#include <stdexcept> // try catch
#include <chrono> // funkcijos darbo laiko fiksavimas
#include <list>
#include <numeric>

#endif
```

6.3 include/studentas.h File Reference

```
#include "lib.h"
```

Classes

- class **Human**

  *Abstract father class for **StudentasClass** (p. 11).*
- class **StudentasClass**

  Student data processing.

Functions

- void **inputManual** (std::vector< **StudentasClass** > &studentai, int studSk)

  Lets the user manually input Student data into a Vector.
- void **inputManualList** (std::list< **StudentasClass** > &studentaiList, int studSk)

  Lets the user manually input Student data into a List.
- void **inputScan** (std::vector< **StudentasClass** > &studentai, std::string failoPav)

  Reads Student data from a File.
- void **outputManual** (**StudentasClass** Lok, int vidMed)

  Outputs Student data from nanual user input and displays the Grade Average or Median.
- void **outputScan** (std::vector< **StudentasClass** > &studentai)

  Outputs Student data into a summary File.
- void **generateEntries** (int studGenSk, int ndGenSk)

  Randomly generates Student data.
- void **inputSplitSort** (std::string failoPav, int rusiavKateg, int useVector, int testStrat)

  Intermediate function for choosing vector or list containers.
- void **inputSplitSortImpl** (std::string failoPav, int rusiavKateg, int testStrat)

  Processes Student data.

6.3.1 Function Documentation

6.3.1.1 generateEntries()

```
void generateEntries (
             int studGenSk,
             int ndGenSk)
```

Randomly generates Student data.

Parameters

studGenSk	Number of students
ndGenSk	Number of homework grades

6.3.1.2 inputManual()

```
void inputManual (
             std::vector< StudentasClass > & studentai,
             int studSk)
```

Lets the user manually input Student data into a Vector.

Parameters

studentai	Vector of Students
studSk	Number of Students

6.3.1.3 inputManualList()

```
void inputManualList (
            std::list< StudentasClass > & studentaiList,
            int studSk)
```

Lets the user manually input Student data into a List.

Parameters

studentaiList	List of Students
studSk	Number of Students

6.3.1.4 inputScan()

```
void inputScan (
            std::vector< StudentasClass > & studentai,
            std::string failoPav)
```

Reads Student data from a File.

Parameters

studentai	
failoPav	

6.3.1.5 inputSplitSort()

```
void inputSplitSort (
            std::string failoPav,
            int rusiavKateg,
            int useVector,
            int testStrat)
```

Intermediate function for choosing vector or list containers.

Parameters

failoPav	Name of file containing Student data
rusiavKateg	Sorting by First name, Last name or Final grade
useVector	
testStrat	Sorting methodology option

6.3.1.6 inputSplitSortImpl()

```
void inputSplitSortImpl (
             std::string failoPav,
             int rusiavKateg,
             int testStrat)
```

Processes Student data.

Template Parameters

Container	Vector or list

Parameters

failoPav	Name of file containing Student data
rusiavKateg	Sorting by First name, Last name or Final grade
testStrat	Sorting methodology option

6.3.1.7 outputManual()

```
void outputManual (
              StudentasClass Lok,
             int vidMed)
```

Outputs Student data from nanual user input and displays the Grade Average or Median.

Parameters

Lok	
vidMed	Grade Average or Median

6.3.1.8 outputScan()

```
void outputScan (
             std::vector< StudentasClass > & studentai)
```

Outputs Student data into a summary File.

Parameters

studentai	

6.4 studentas.h

Go to the documentation of this file.

```
#ifndef STUDENTAS_H_INCLUDED
#define STUDENTAS_H_INCLUDED

#include "lib.h"

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

public:
    virtual ~Human() = 0;

    // Visiškai virtualūs metodai
    virtual void setVardas(const std::string& vardas) = 0;
    virtual void setPavarde(const std::string& pavarde) = 0;

    virtual const std::string& getVardas() const = 0;
    virtual const std::string& getPavarde() const = 0;
};

inline Human::~Human() {}

class StudentasClass : public Human{
private:
    std::vector<int> tarpRez_;
    int egzamRez_;
    double vidurkis_;
    double mediana_;
    double galutinis_;
    bool islaike_;

    // Privatūs pagalbiniai metodai
    void rastiVid();
    void rastiMed();
    void rastiRez();

public:
    // Default konstruktorius
    StudentasClass() :
        Human(),
        egzamRez_(0),
        vidurkis_(0.0),
        mediana_(0.0),
        galutinis_(0.0),
        islaike_(false) {}

    // Parametrizuotas konstruktorius
    StudentasClass(const std::string& vardas, const std::string& pavarde) :
        Human(),
        egzamRez_(0),
        vidurkis_(0.0),
        mediana_(0.0),
        galutinis_(0.0),
        islaike_(false) {
            this->setVardas(vardas);
            this->setPavarde(pavarde);
        }

    ~StudentasClass(){
        tarpRez_.clear();
    };

    // Copy konstruktorius
    StudentasClass(const StudentasClass& other) : Human() {
        this->setVardas(other.getVardas());
        this->setPavarde(other.getPavarde());
        tarpRez_ = other.tarpRez_;
        egzamRez_ = other.egzamRez_;
        vidurkis_ = other.vidurkis_;
        mediana_ = other.mediana_;
        galutinis_ = other.galutinis_;
        islaike_ = other.islaike_;
    }

    // Copy assignment konstruktorius
    StudentasClass& operator=(const StudentasClass& other) {
        if (this != &other) {  // Apsauga nuo savęs priskyrimo
            vardas_ = other.vardas_;
            pavarde_ = other.pavarde_;
            tarpRez_ = other.tarpRez_;
            egzamRez_ = other.egzamRez_;
            vidurkis_ = other.vidurkis_;
            mediana_ = other.mediana_;
            galutinis_ = other.galutinis_;
            islaike_ = other.islaike_;
        }
        return *this;
    }

    // Get’eriai
    const std::string& getVardas() const override { return vardas_; }
    const std::string& getPavarde() const override { return pavarde_; }
    const std::vector<int>& getTarpRez() const { return tarpRez_; }
    int getEgzamRez() const { return egzamRez_; }
    double getVidurkis() const { return vidurkis_; }
    double getMediana() const { return mediana_; }
    double getGalutinis() const { return galutinis_; }

    // Set’eriai
    void setVardas(const std::string& vardas) override { vardas_ = vardas; }
    void setPavarde(const std::string& pavarde) override { pavarde_ = pavarde; }
    void setEgzamRez(int rez) { egzamRez_ = rez; rastiGalutini(); }
    void setTarpRez(const std::vector<int>& naujiTarpRez);

    // Metodai
    void pridetiTarpRez(int rez);
    void generuotiBalus(int kiekBalu = 15);
    void rastiGalutini(bool naudotiVidurki = true);
    void isimtiGalutini();
    void clear();
    void rastiIslaike(){ islaike_ = (galutinis_ >= 5.0); }
    bool arIslaike() const { return islaike_; }
    bool compare(const StudentasClass& b, int criteria = 2);
    std::ostream& printInfo(std::ostream& os) const;

    //Friend’ai įvedimo ir išvedimo operacijoms
    friend std::istream& operator»(std::istream& is, StudentasClass& s);
    friend std::ostream& operator«(std::ostream& os, const StudentasClass& s);
};


// Pre-v1.0, ne member metodai darbui su klase Studentas
void inputManual(std::vector<StudentasClass> &studentai, int studSk);
void inputManualList(std::list<StudentasClass> &studentaiList, int studSk);
void inputScan(std::vector<StudentasClass> &studentai, std::string failoPav);
void outputManual(StudentasClass Lok, int vidMed);
void outputScan(std::vector<StudentasClass> &studentai);
void generateEntries(int studGenSk, int ndGenSk);
void inputSplitSort(std::string failoPav, int rusiavKateg, int useVector, int testStrat);
void inputSplitSortImpl(std::string failoPav, int rusiavKateg, int testStrat);

#endif
```

6.5 README.md File Reference

6.6 src/failuGeneratorius.cpp File Reference

```
#include "lib.h"
#include "studentas.h"
```

Classes

- class **Timer**

  Tracks time for run-time measurements.

Functions

- void **generateEntries** (int studGenSk, int ndGenSk)

  Randomly generates Student data.
- template<typename Container >
  void **inputSplitSortImpl** (std::string failoPav, int rusiavKateg, int testStrat)
- void **inputSplitSort** (std::string failoPav, int rusiavKateg, int useVector, int testStrat)

  Intermediate function for choosing vector or list containers.

6.6.1 Function Documentation

6.6.1.1 generateEntries()

```
void generateEntries (
            int studGenSk,
            int ndGenSk)
```

Randomly generates Student data.

Parameters

studGenSk	Number of students
ndGenSk	Number of homework grades

6.6.1.2 inputSplitSort()

```
void inputSplitSort (
            std::string failoPav,
            int rusiavKateg,
            int useVector,
            int testStrat)
```

Intermediate function for choosing vector or list containers.

Parameters

failoPav	Name of file containing Student data
rusiavKateg	Sorting by First name, Last name or Final grade
useVector	
testStrat	Sorting methodology option

6.6.1.3 inputSplitSortImpl()

```
template<typename Container >
void inputSplitSortImpl (
            std::string failoPav,
            int rusiavKateg,
            int testStrat)
```

6.7 src/main.cpp File Reference

```
#include "lib.h"
#include "studentas.h"
```

Functions

- int **main** ()

6.7.1 Function Documentation

6.7.1.1 main()

```
int main ()
```

6.8 src/studentas.cpp File Reference

```
#include "lib.h"
#include "studentas.h"
```

Functions

- std::istream & **operator>>** (std::istream &is, **StudentasClass** &s)
- std::ostream & **operator<<** (std::ostream &os, const **StudentasClass** &s)
- void **inputManual** (std::vector< **StudentasClass** > &studentai, int studSk)

  Lets the user manually input Student data into a Vector.
- void **inputManualList** (std::list< **StudentasClass** > &studentaiList, int studSk)

  Lets the user manually input Student data into a List.
- void **inputScan** (std::vector< **StudentasClass** > &studentai, std::string failoPav)

  Reads Student data from a File.
- void **outputManual** (**StudentasClass** stud, int vidMed)

  Outputs Student data from nanual user input and displays the Grade Average or Median.
- void **outputScan** (std::vector< **StudentasClass** > &studentai)

  Outputs Student data into a summary File.

6.8.1 Function Documentation

6.8.1.1 inputManual()

```
void inputManual (
            std::vector< StudentasClass > & studentai,
            int studSk)
```

Lets the user manually input Student data into a Vector.

Parameters

studentai	Vector of Students
studSk	Number of Students

6.8.1.2 inputManualList()

```
void inputManualList (
            std::list< StudentasClass > & studentaiList,
            int studSk)
```

Lets the user manually input Student data into a List.

Parameters

studentaiList	List of Students
studSk	Number of Students

6.8.1.3 inputScan()

```
void inputScan (
            std::vector< StudentasClass > & studentai,
            std::string failoPav)
```

Reads Student data from a File.

Parameters

studentai	
failoPav	

6.8.1.4 operator<<()

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

6.8.1.5 operator>>()

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

6.8.1.6 outputManual()

```
void outputManual (
             StudentasClass Lok,
            int vidMed)
```

Outputs Student data from nanual user input and displays the Grade Average or Median.

Parameters

Lok	
vidMed	Grade Average or Median

6.8.1.7 outputScan()

```
void outputScan (
            std::vector< StudentasClass > & studentai)
```

Outputs Student data into a summary File.

Parameters

studentai	

Index