Schedule Management System

Generated by Doxygen 1.9.4

Chapter 1

Schedule Management System

Project made by:

- Henrique Fernandes 202204988
- José Sousa 202208817
- Leandro Martins 202208001

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 documented files with brief descriptions:

Chapter 5

Class Documentation

5.1 AddRequest Class Reference

Request of type add.

`#include <AddRequest.h>`

Public Member Functions

- AddRequest (const std::string &studentID, const std::string &ucCode, const std::string &classCode)
  Parameterized constructor.
- std::string getStudentID () const
  Gets the student ID.
- std::string getUCCode () const
  Gets the course code.
- std::string getClassCode () const
  Gets the class code.
- void setStudentID (const std::string &studentID)
  Sets a new student ID.
- void setUCCode (const std::string &ucCode)
  Sets a new couse code.
- void setClassCode (const std::string &classCode)
  Sets a new class code.

Additional Inherited Members

5.1.1 Detailed Description

Request of type add.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 AddRequest()

```
AddRequest::AddRequest (
            const std::string & studentID,
            const std::string & ucCode,
            const std::string & classCode )
```

Parameterized constructor.

Parameters

studentID	String representing the student ID.
ucCode	String representing the course code.
classCode	String representing the class code.

5.1.3 Member Function Documentation

5.1.3.1 getClassCode()

```
std::string AddRequest::getClassCode ( ) const
```

Gets the class code.

Returns

A string representing the class code.

5.1.3.2 getStudentID()

```
std::string AddRequest::getStudentID ( ) const
```

Gets the student ID.

Returns

A string representing the studentID.

5.1.3.3 getUCCode()

```
std::string AddRequest::getUCCode ( ) const
```

Gets the course code.

Returns

A string representing the course code.

5.1.3.4 setClassCode()

```
void AddRequest::setClassCode (
            const std::string & classCode )
```

Sets a new class code.

Parameters

classCode	The new class code.

5.1.3.5 setStudentID()

```
void AddRequest::setStudentID (
            const std::string & studentID )
```

Sets a new student ID.

Parameters

studentID	The new student ID.

5.1.3.6 setUCCode()

```
void AddRequest::setUCCode (
            const std::string & ucCode )
```

Sets a new couse code.

Parameters

ucCode	The new course code.

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

- src/AddRequest.h
- src/AddRequest.cpp

5.2 ControlUnit Class Reference

Class used to handle the core functions of the program.

`#include <ControlUnit.h>`

Public Member Functions

- void loadCSV (string studentFilename)

  Loads all the csv files.
- void **LoadClassesCSV** ()

  Loads the classes.csv file (which has all the lessons).
- void **LoadClassesPerUcCSV** ()

  Loads the classes_per_uc.csv file (which has all the courses and classes).
- void **LoadStudentsClassesCSV** ()

  Load students_classes.csv or student_classes_updated.csv, depending on the option chose.
- void **saveChanges** ()

  Saves the changes made, updating the file students_classes_updated.csv.
- void **DisplayStudentSchedule** ()

  Displays the schedule of a student.
- void **DisplayClassSchedule** ()

  Displays the schedule of a class.
- int StudentsInAtLeastNUcs (int n)

  Displays the students enrolled in at least N courses.
- int StudentsInAtMostNUcs (int n)

  Displays the students enrolled in at most N courses.
- int StudentsInUcs (int n)

  Displays the students enrolled in exactly N courses.
- void courseStudents (string courseCode, function< bool(Student, Student)> func)

  Displays the students enrolled in a specific course.
- void yearStudents (char year, function< bool(Student, Student)> func)

  Displays the students from a specific year.
- void classStudents (string classCode, function< bool(Student, Student)> func)

  Displays the students from a specific class.
- void **UCWithMostStudents** ()

  Displays all the courses starting with the one with the most student.
- int NumBalanced (vector< studentGroup >, map< MainKey, int >)

  Checks the balance of the classes.
- bool IsThereConflict (vector< lesson >)

  Detects conflicts in a schedule.
- bool processRequest (Request *request, bool bypassStack=false)

  Processes a request.
- void processAddRequest (AddRequest *addRequest)

Processes a request of type add.

- void processRemoveRequest (RemoveRequest *removeRequest)

  Processes a request of type remove.
- void processSwitchRequest (SwitchRequest *switchRequest)

  Processes a request of type switch.
- void **processAllRequests** ()

  Processes all the requests awaiting to be processed.
- void **removeLastPendingRequest** ()

  Removes the most recent request that hasn't been applied.
- void undoRequest (int n)

  Undoes the N most recent applied request.
- void **createAdd** ()

  Creates a request of type add.
- void **createRemove** ()

  Creates a request of type remove.
- void **createSwitch** ()

  Creates a request of type switch.
- bool CheckAdd (AddRequest *addrq)

  Checks if the request is possible.
- bool CheckRemove (RemoveRequest *remrq)

  Checks if the request is possible.
- bool CheckSwitch (SwitchRequest *swrq)

  Checks if the request is possible.
- string getClassInUc (string studentID, string ucCode)

  Gets the class of a student knowing the course.
- void **clearMemory** ()

  Frees all the dynamic memory.

5.2.1 Detailed Description

Class used to handle the core functions of the program.

5.2.2 Member Function Documentation

5.2.2.1 CheckAdd()

```
bool ControlUnit::CheckAdd (
            AddRequest * addrq )
```

Checks if the request is possible.

Parameters

addrq	Request to be analysed.

Returns

Boolean representing if the request is possible or not.

5.2.2.2 CheckRemove()

```
bool ControlUnit::CheckRemove (
            RemoveRequest * remrq )
```

Checks if the request is possible.

Parameters

remrq	Request to be analysed.

Returns

Boolean representing if the request is possible or not.

5.2.2.3 CheckSwitch()

```
bool ControlUnit::CheckSwitch (
            SwitchRequest * swrq )
```

Checks if the request is possible.

Parameters

swrq	Request to be analysed.

Returns

Boolean representing if the request is possible or not.

5.2.2.4 classStudents()

```
void ControlUnit::classStudents (
            string classCode,
            function< bool(Student, Student)> func )
```

Displays the students from a specific class.

Parameters

classCode	String representing the class code.
func	Boolean function used to sort students.

5.2.2.5 courseStudents()

```
void ControlUnit::courseStudents (
            string courseCode,
            function< bool(Student, Student)> func )
```

Displays the students enrolled in a specific course.

Parameters

courseCode	String representing the course code.
func	Boolean function used to sort students.

5.2.2.6 getClassInUc()

```
string ControlUnit::getClassInUc (
            string studentID,
            string ucCode )
```

Gets the class of a student knowing the course.

Parameters

studentID	String representing the student ID.
ucCode	String representing the course code.

Returns

String representing the class code.

5.2.2.7 IsThereConflict()

```
bool ControlUnit::IsThereConflict (
            vector< lesson > lessons )
```

Detects conflicts in a schedule.

Returns

Boolean that represents the existence of conflicts.

5.2.2.8 loadCSV()

```
void ControlUnit::loadCSV (
            string studentFilename )
```

Loads all the csv files.

Parameters

studentFilename	A string that represents the student csv, it can eitheir be the original version or the updated version.

5.2.2.9 NumBalanced()

```
int ControlUnit::NumBalanced (
            vector< studentGroup > groups,
            map< MainKey, int > myMap )
```

Checks the balance of the classes.

Returns

Returns the maximum difference between the amount of students in each class.

5.2.2.10 processAddRequest()

```
void ControlUnit::processAddRequest (
            AddRequest * addRequest )
```

Processes a request of type add.

Parameters

addRequest	The request to be processed.

5.2.2.11 processRemoveRequest()

```
void ControlUnit::processRemoveRequest (
            RemoveRequest * removeRequest )
```

Processes a request of type remove.

Parameters

removeRequest	The request to be processed.

5.2.2.12 processRequest()

```
bool ControlUnit::processRequest (
            Request * request,
            bool bypassStack = false )
```

Processes a request.

Parameters

request	Request to be processed.
bypassStack	A boolean that states if the request should bypass the stack (true if the request is an undo of a previous request).

Returns

Boolean that represents if the request was processed successfully.

5.2.2.13 processSwitchRequest()

```
void ControlUnit::processSwitchRequest (
            SwitchRequest * switchRequest )
```

Processes a request of type switch.

Parameters

switchRequest	The request to be processed.

5.2.2.14 StudentsInAtLeastNUcs()

```
int ControlUnit::StudentsInAtLeastNUcs (
            int n )
```

Displays the students enrolled in at least N courses.

Parameters

n	Integer representing the minimum amount of courses.

Returns

Integer representing the amount of students enrolled in at least N courses.

5.2.2.15 StudentsInAtMostNUcs()

```
int ControlUnit::StudentsInAtMostNUcs (
            int n )
```

Displays the students enrolled in at most N courses.

Parameters

n	Integer representing the maximum amount of courses.

Returns

Integer representing the amount of students enrolled in at most N courses.

5.2.2.16 StudentsInUcs()

```
int ControlUnit::StudentsInUcs (
            int n )
```

Displays the students enrolled in exactly N courses.

Parameters

n	Integer representing the amount of courses.

Returns

Integer representing the amount of students enrolle in N courses.

5.2.2.17 undoRequest()

```
void ControlUnit::undoRequest (
            int n )
```

Undoes the N most recent applied request.

Parameters

n	Integer representing how many requests should be undone.

5.2.2.18 yearStudents()

```
void ControlUnit::yearStudents (
            char year,
            function< bool(Student, Student)> func )
```

Displays the students from a specific year.

Parameters

year	Char representing the year,
func	Boolean function used to sort students.

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

- src/ControlUnit.h
- src/ControlUnit.cpp

5.3 lesson Class Reference

Class used to represent a lesson from a course.

```
#include <lesson.h>
```

Public Member Functions

- lesson (const std::string &ucCode, const std::string &studentGroup, const std::string &weekday, double startTime, double duration, const std::string &type)

  Parameterized Constructor.
- const std::string & getWeekday () const

  Gets the lesson's weekday.

- const lessontime & getStartTime () const

  Gets the time the lesson starts.
- const lessontime & getDuration () const

  Gets the duration of the lesson.
- const lessontime & getEndTime () const

  Gets the time the lesson ends.
- const string & getUccode () const

  Gets the course code of the lesson.
- const std::string & getType () const

  Gets the type of the lesson.

5.3.1 Detailed Description

Class used to represent a lesson from a course.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 lesson()

```
lesson::lesson (
            const std::string & ucCode,
            const std::string & studentGroup,
            const std::string & weekday,
            double startTime,
            double duration,
            const std::string & type )
```

Parameterized Constructor.

Parameters

ucCode	String representing the course.
studentGroup	String representing the class.
weekday	String representing the weekday.
startTime	The time the lesson starts.
duration	The duration of the lesson.
type	The type of the lesson.

5.3.3 Member Function Documentation

5.3.3.1 getDuration()

```
const lessontime & lesson::getDuration ( ) const
```

Gets the duration of the lesson.

Returns

The duration of the lesson.

5.3.3.2 getEndTime()

```
const lessontime & lesson::getEndTime ( ) const
```

Gets the time the lesson ends.

Returns

The time the lesson ends.

5.3.3.3 getStartTime()

```
const lessontime & lesson::getStartTime ( ) const
```

Gets the time the lesson starts.

Returns

The time the lesson starts.

5.3.3.4 getType()

```
const std::string & lesson::getType ( ) const
```

Gets the type of the lesson.

Returns

A string representing the type of the lesson.

5.3.3.5 getUccode()

```
const std::string & lesson::getUccode ( ) const
```

Gets the course code of the lesson.

Returns

A string representing the course code.

5.3.3.6 getWeekday()

```
const std::string & lesson::getWeekday ( ) const
```

Gets the lesson's weekday.

Returns

A string representing the weekday.

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

- src/lesson.h
- src/lesson.cpp

5.4 lessontime Class Reference

Class used to represent time.

```
#include <lessontime.h>
```

Public Member Functions

- lessontime (double time)
  Copy constructor.
- **lessontime** ()
  Default constructor (00:00)
- lessontime (int hour, int minutes)
  Parameterized constructor.
- string displayHourFormat () const
  Converts the time to a string.
- int getHour () const
  Hour getter.
- int getMinute () const
  Minutes getter.

5.4.1 Detailed Description

Class used to represent time.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 lessontime() [1/2]

```
lessontime::lessontime (
            double time )  [explicit]
```

Copy constructor.

Parameters

time	

5.4.2.2 lessontime() [2/2]

```
lessontime::lessontime (
            int hour,
            int minutes )
```

Parameterized constructor.

Parameters

hour	
minutes	

5.4.3 Member Function Documentation

5.4.3.1 displayHourFormat()

```
std::string lessontime::displayHourFormat ( ) const
```

Converts the time to a string.

Returns

A string representing the time.

5.4.3.2 getHour()

```
int lessontime::getHour ( ) const
```

Hour getter.

Returns

An integer representing the hour.

5.4.3.3 getMinute()

```
int lessontime::getMinute ( ) const
```

Minutes getter.

Returns

An integer representing the minutes.

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

- src/lessontime.h
- src/lessontime.cpp

5.5 Menu Class Reference

Class used to represent the menu the user uses to navigate.

```
#include <Menu.h>
```

Public Member Functions

- void **createMenu** ()
  Creates the menu.
- void **SeeStudentSchedule** ()
  Displays the schedule of a student.
- void **SeeClassSchedule** ()
  Displays the schedule of a class.
- void **SeeNumStudentsInExactNUCs** ()
  Displays the student enrolled in exactly N courses.
- void **SeeNumStudentsAtLeastNUCs** ()
  Displays the students enrolled in at least N courses.
- void **SeeNumStudentsAtMostNUCs** ()
  Displays the student enrolled in at most N courses.
- void **SeeUcFromMostStudents** ()

Displays all the courses starting with the one with the most student.

- void **SeeNumStudentsInNUCs** ()

  Enters the submenu for listing the students in courses.
- void **listingMenu** ()

  Enters the listing menu, which allows the user to list students, see schedules etc.
- void **requestMenu** ()

  Enters the request menu, which allows the user to create, delete and manage requests.
- void **scheduleMenu** ()

  Enters the schedule menu, which allows the user to see the schedule for a student or a class.
- void **studentMenu** ()

  Enters the student menu, which allows the user to see all students from a year, course or class.
- void SeeStudentsInUc (function< bool(Student, Student)> comp)

  Lists all the students in a specific course.
- void SeeStudentsInYear (function< bool(Student, Student)> comp)

  Lists all the students in a specific year.
- void SeeStudentsInClass (function< bool(Student, Student)> comp)

  Lists all the students in a specific class.
- void **createRequest** ()

  Enters the menu for creating request, allowing users to add, remove or switch classes.
- function< bool(Student, Student)> optionStudentMenu ()

  Allows the user to select different sorting options for displaying the students.

5.5.1 Detailed Description

Class used to represent the menu the user uses to navigate.

5.5.2 Member Function Documentation

5.5.2.1 optionStudentMenu()

```
function< bool(Student, Student)> Menu::optionStudentMenu ( )
```

Allows the user to select different sorting options for displaying the students.

Returns

A boolean function the compares students.

5.5.2.2 SeeStudentsInClass()

```
void Menu::SeeStudentsInClass (
            function< bool(Student, Student)> comp )
```

Lists all the students in a specific class.

Parameters

comp	A boolean function that compares the students, allowing the program to list the students in different ways.

5.5.2.3 SeeStudentsInUc()

```
void Menu::SeeStudentsInUc (
            function< bool(Student, Student)> comp )
```

Lists all the students in a specific course.

Parameters

comp	A boolean function that compares the students, allowing the program to list the students in different ways.

5.5.2.4 SeeStudentsInYear()

```
void Menu::SeeStudentsInYear (
            function< bool(Student, Student)> comp )
```

Lists all the students in a specific year.

Parameters

comp	A boolean function that compares the students, allowing the program to list the students in different ways.

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

- src/Menu.h
- src/Menu.cpp

5.6 RemoveRequest Class Reference

Request of type remove.

```
#include <RemoveRequest.h>
```

Public Member Functions

- RemoveRequest (const std::string &studentID, const std::string &ucCode, const std::string &classCode)

  Parameterized constructor.
- std::string getStudentID () const

  Gets the student ID.
- std::string getUCCode () const

  Gets the course code.
- std::string getClassCode () const

  Gets the class code.
- void setStudentID (const std::string &studentID)

  Sets a new student ID.
- void setUCCode (const std::string &ucCode)

  Sets a new couse code.
- void setClassCode (const std::string &classCode)

  Sets a new class code.

Additional Inherited Members

5.6.1 Detailed Description

Request of type remove.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 RemoveRequest()

```
RemoveRequest::RemoveRequest (
            const std::string & studentID,
            const std::string & ucCode,
            const std::string & classCode )
```

Parameterized constructor.

Parameters

studentID	String representing the student ID.
ucCode	String representing the course code.
classCode	String representing the class code.

5.6.3 Member Function Documentation

5.6.3.1 getClassCode()

```
std::string RemoveRequest::getClassCode ( ) const
```

Gets the class code.

Returns

A string representing the class code.

5.6.3.2 getStudentID()

```
std::string RemoveRequest::getStudentID ( ) const
```

Gets the student ID.

Returns

A string representing the studentID.

5.6.3.3 getUCCode()

```
std::string RemoveRequest::getUCCode ( ) const
```

Gets the course code.

Returns

A string representing the course code.

5.6.3.4 setClassCode()

```
void RemoveRequest::setClassCode (
            const std::string & classCode )
```

Sets a new class code.

Parameters

classCode	The new class code.

5.6.3.5 setStudentID()

```
void RemoveRequest::setStudentID (
            const std::string & studentID )
```

Sets a new student ID.

Parameters

studentID	The new student ID.

5.6.3.6 setUCCode()

```
void RemoveRequest::setUCCode (
            const std::string & ucCode )
```

Sets a new couse code.

Parameters

ucCode	The new course code.

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

- src/RemoveRequest.h
- src/RemoveRequest.cpp

5.7 Request Class Reference

Class used to represent a generic request.

```
#include <Request.h>
```

Public Member Functions

- Request (std::string type)
  Parameterized constructor.
- std::string getType () const
  Gets the type of the request.

Static Public Member Functions

- static void **resetCount** ()
  Resets the requeset counter.

5.7.1 Detailed Description

Class used to represent a generic request.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 Request()

```
Request::Request (
            std::string type )  [inline]
```

Parameterized constructor.

Parameters

type	String representing the request type (add/remove/switch).

5.7.3 Member Function Documentation

5.7.3.1 getType()

```
std::string Request::getType ( ) const  [inline]
```

Gets the type of the request.

Returns

A string representing the type of the request.

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

- src/Request.h

5.8 Schedule Class Reference

Class used to display a schedule.

```
#include <Schedule.h>
```

Public Member Functions

- Schedule (vector< lesson > lessons)

  Parameterized constructor.
- void **display** ()

  Displays the schedule.

5.8.1 Detailed Description

Class used to display a schedule.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 Schedule()

```
Schedule::Schedule (
            vector< lesson > lessons )
```

Parameterized constructor.

Parameters

lessons	Vector with all the lessons to be displayed.

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

- src/Schedule.h
- src/Schedule.cpp

5.9 Student Class Reference

Class used to represent a student.

```
#include <student.h>
```

Public Member Functions

- **Student** ()=default

  Default constructor.
- Student (string studentId, string name, set< studentGroup > group)

  Parameterized constructor.
- std::string getStudentID () const

  Gets the student ID.

- set< studentGroup > getStudentGroups () const

  Gets all the classes the student belongs to.
- std::string getName () const

  Gets the name of the student.
- void setName (const std::string &newName)

  Sets the newName of the student.
- void setStudentID (const std::string &studentId)

  Sets the student ID.
- void addStudentGroup (const studentGroup &GroupToAdd)

  Adds a new class to the student.
- void removeGroup (const studentGroup &GroupToRemove)

  Removes a class from the student.
- bool isInUC (const string &uc) const

  Detects if the student is enrolled in a certain course.
- bool isInClass (const string &ucCode, const string &studentGroup) const

  Detects if the student is enrolled in a certain class from a couse.

5.9.1 Detailed Description

Class used to represent a student.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 Student()

```
Student::Student (
            string studentId,
            string name,
            set< studentGroup > group )
```

Parameterized constructor.

Parameters

studentId	String representing the student ID.
name	String representing the name of the student.
group	A set with the classes the student has.

5.9.3 Member Function Documentation

5.9.3.1 addStudentGroup()

```
void Student::addStudentGroup (
            const studentGroup & GroupToAdd )
```

Adds a new class to the student.

Parameters

GroupToAdd	

5.9.3.2 getName()

```
string Student::getName ( ) const
```

Gets the name of the student.

Returns

A string representing the name of the student.

5.9.3.3 getStudentGroups()

```
set< studentGroup > Student::getStudentGroups ( ) const
```

Gets all the classes the student belongs to.

Returns

A set of classes that the student belongs to.

5.9.3.4 getStudentID()

```
string Student::getStudentID ( ) const
```

Gets the student ID.

Returns

A string representing the student ID.

5.9.3.5 isInClass()

```
bool Student::isInClass (
            const string & ucCode,
            const string & studentGroup ) const
```

Detects if the student is enrolled in a certain class from a couse.

Parameters

ucCode	String representing a couse.
studentGroup	String representing a class.

Returns

Returns true if the student is enrolled in a certain class form a course.

5.9.3.6 isInUC()

```
bool Student::isInUC (
            const string & uc ) const
```

Detects if the student is enrolled in a certain course.

Parameters

uc	String representing a course.

Returns

Returns true if the student is enrolled in a certain course.

5.9.3.7 removeGroup()

```
void Student::removeGroup (
            const studentGroup & GroupToRemove )
```

Removes a class from the student.

Parameters

GroupToRemove	

5.9.3.8 setName()

```
void Student::setName (
            const std::string & newName )
```

Sets the newName of the student.

Parameters

newName	A string representing the newName of the student

5.9.3.9 setStudentID()

```
void Student::setStudentID (
            const std::string & studentId )
```

Sets the student ID.

Parameters

studentId	A string representing the new student ID.

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

- src/student.h
- src/student.cpp

5.10 studentGroup Class Reference

Class used to represent a class (group of students).

```
#include <studentGroup.h>
```

Public Member Functions

- **studentGroup** ()=default

  Default constructor.
- studentGroup (const std::string &ucCode, const std::string &classCode)

  Parameterized constructor.
- const std::string & getClassCode () const

  Gets the class code.
- const std::string & getUcCode () const

  Gets the course code.

5.10.1 Detailed Description

Class used to represent a class (group of students).

5.10.2 Constructor & Destructor Documentation

5.10.2.1 studentGroup()

```
studentGroup::studentGroup (
            const std::string & ucCode,
            const std::string & classCode )
```

Parameterized constructor.

Parameters

ucCode	String representing the course code.
classCode	String representing the class code.

5.10.3 Member Function Documentation

5.10.3.1 getClassCode()

```
const std::string & studentGroup::getClassCode ( ) const  [inline]
```

Gets the class code.

Returns

A string representing the class code.

5.10.3.2 getUcCode()

```
const std::string & studentGroup::getUcCode ( ) const  [inline]
```

Gets the course code.

Returns

A string representing the course code.

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

- src/studentGroup.h
- src/studentGroup.cpp

5.11 SwitchRequest Class Reference

Request of type switch.

`#include <SwitchRequest.h>`

Public Member Functions

- SwitchRequest (const std::string &studentID, const std::string &ucCode1, const std::string &ucCode2, const std::string &classCode1, const std::string &classCode2)

  Parameterized constructor.
- std::string getStudentID () const

  Gets the student ID.
- std::string getUCCode1 () const

  Gets the current course code.
- std::string getUCCode2 () const

  Gets the new course code.
- std::string getClassCode1 () const

  Gets the current class code.
- std::string getClassCode2 () const

  Gets the new class code.

Additional Inherited Members

5.11.1 Detailed Description

Request of type switch.

5.11.2 Constructor & Destructor Documentation

5.11.2.1 SwitchRequest()

```
SwitchRequest::SwitchRequest (
            const std::string & studentID,
            const std::string & ucCode1,
            const std::string & ucCode2,
            const std::string & classCode1,
            const std::string & classCode2 )
```

Parameterized constructor.

Parameters

studentID	String representing the student ID.
ucCode1	String representing the current course code.
ucCode2	String representing the new course code.
classCode1	String representing the current class code.
classCode2	String representing the new class code.

5.11.3 Member Function Documentation

5.11.3.1 getClassCode1()

```
std::string SwitchRequest::getClassCode1 ( ) const
```

Gets the current class code.

Returns

A string representing the current class code.

5.11.3.2 getClassCode2()

```
std::string SwitchRequest::getClassCode2 ( ) const
```

Gets the new class code.

Returns

A string representing the new class code.

5.11.3.3 getStudentID()

```
std::string SwitchRequest::getStudentID ( ) const
```

Gets the student ID.

Returns

A string representing the student ID.

5.11.3.4 getUCCode1()

```
std::string SwitchRequest::getUCCode1 ( ) const
```

Gets the current course code.

Returns

A string representing the current course code.

5.11.3.5 getUCCode2()

```
std::string SwitchRequest::getUCCode2 ( ) const
```

Gets the new course code.

Returns

A string representing the new course code.

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

- src/SwitchRequest.h
- src/SwitchRequest.cpp

Chapter 6

File Documentation

6.1 src/AddRequest.h File Reference

```
#include <string>
#include "Request.h"
```

Classes

- class AddRequest

  Request of type add.

6.2 AddRequest.h

Go to the documentation of this file.

```
1
2 #ifndef PROJAED_ADDREQUEST_H
3 #define PROJAED_ADDREQUEST_H
4
5 #include <string>
6 #include "Request.h"
7
11 class AddRequest :  public Request {
12 private:
13     std::string studentID;
14     std::string ucCode;
15     std::string classCode;
16
17
18 public:
25     AddRequest(const std::string &studentID, const std::string &ucCode, const std::string &classCode);
26
31     std::string getStudentID() const;
32
37     std::string getUCCode() const;
38
43     std::string getClassCode() const;
44
49     void setStudentID(const std::string &studentID);
50
55     void setUCCode(const std::string &ucCode);
56
61     void setClassCode(const std::string &classCode);
62 };
63
64 #endif //PROJAED_ADDREQUEST_H
```

6.3 src/ControlUnit.h File Reference

```
#include <vector>
#include <string>
#include "studentGroup.h"
#include <map>
#include "student.h"
#include <set>
#include <list>
#include <queue>
#include <stack>
#include <functional>
#include "lesson.h"
#include "Request.h"
#include "AddRequest.h"
#include "RemoveRequest.h"
#include "SwitchRequest.h"
```

Classes

- class ControlUnit

  Class used to handle the core functions of the program.

6.4 ControlUnit.h

Go to the documentation of this file.

```
1
2 #ifndef PROJAED_CONTROLUNIT_H
3 #define PROJAED_CONTROLUNIT_H
4
5
6 #include <vector>
7 #include <string>
8 #include "studentGroup.h"
9 #include <map>
10 #include "student.h"
11 #include <set>
12 #include <list>
13 #include <queue>
14 #include <stack>
15 #include <functional>
16 #include "lesson.h"
17 #include "Request.h"
18 #include "AddRequest.h"
19 #include "RemoveRequest.h"
20 #include "SwitchRequest.h"
21
25 class ControlUnit {
26
27 private:
28     struct MainKey {
29         string ucCode;
30         string ClassCode;
31
32         bool operator<(const MainKey &other)const {
33             if (ucCode != other.ucCode) {
34                 return ucCode < other.ucCode;
35             }
36             return ClassCode < other.ClassCode;
37         }
38     };
39
40     string filename;
41     set<Student> StudentSet;
42     vector<lesson> LessonVector;
```

```
43     list<studentGroup> StudentGroupList;
44     map<MainKey, studentGroup *> KeyToStudentGroup;
45     map<MainKey, set<lesson *» LessonMap;
46     map<MainKey, int> SizeMap;
47     queue<Request *> RequestsToProcess;
48     stack<Request *> ProcessedRequests;
49     int cap = 30;
50
51 public :
52
57     void loadCSV(string studentFilename);
58
62     void LoadClassesCSV();
63
67     void LoadClassesPerUcCSV();
68
72     void LoadStudentsClassesCSV();
73
77     void saveChanges();
78
82     void DisplayStudentSchedule();
83
87     void DisplayClassSchedule();
88
94     int StudentsInAtLeastNUcs(int n);
95
101     int StudentsInAtMostNUcs(int n);
102
108     int StudentsInUcs(int n);
109
115     void courseStudents(string courseCode, function<bool(Student, Student)> func);
116
122     void yearStudents(char year, function<bool(Student, Student)> func);
123
129     void classStudents(string classCode, function<bool(Student, Student)> func);
130
134     void UCWithMostStudents();
135
140     int NumBalanced(vector<studentGroup>, map<MainKey, int>);
141
146     bool IsThereConflict(vector<lesson>);
147
154     bool processRequest(Request *request, bool bypassStack = false);
155
160     void processAddRequest(AddRequest *addRequest);
161
166     void processRemoveRequest(RemoveRequest *removeRequest);
167
172     void processSwitchRequest(SwitchRequest *switchRequest);
173
177     void processAllRequests();
178
182     void removeLastPendingRequest();
183
188     void undoRequest(int n); //this method removes last n applied request
189
190     void CheckIfThereAreConflicts();
191
195     void createAdd();
196
200     void createRemove();
201
205     void createSwitch();
206
212     bool CheckAdd(AddRequest *addrq);
213
219     bool CheckRemove(RemoveRequest *remrq);
220
226     bool CheckSwitch(SwitchRequest *swrq);
227
234     string getClassInUc(string studentID, string ucCode);
235
239     void clearMemory();
240 };
241
242
243 #endif //PROJAED_CONTROLUNIT_H
```

6.5 src/lesson.h File Reference

```
#include <string>
#include <ctime>
```

```
#include "lessontime.h"
#include <iostream>
#include <map>
```

Classes

- class lesson

  Class used to represent a lesson from a course.

6.6 lesson.h

Go to the documentation of this file.

```
1
2 #ifndef PROJAED_LESSON_H
3 #define PROJAED_LESSON_H
4
5
6 #include <string>
7 #include <ctime>
8 #include "lessontime.h"
9 #include <iostream>
10 #include <map>
11
15 class lesson {
16 public:
26     lesson(const std::string &ucCode, const std::string &studentGroup, const std::string &weekday, double
       startTime,
27            double duration, const std::string &type);
28
33     const std::string &getWeekday() const;
34
39     const lessontime &getStartTime() const;
40
45     const lessontime &getDuration() const;
46
51     const lessontime &getEndTime() const;
52
57     const string &getUccode() const;
58
63     const std::string &getType() const;
64
65     friend std::ostream &operator«(std::ostream &os, const lesson &lesson);
66     mutable std::map<std::string, int> dayMap = {{"Monday",    0},
67                                                  {"Tuesday",   1},
68                                                  {"Wednesday", 2},
69                                                  {"Thursday",  3},
70                                                  {"Friday",    4},
71                                                  {"Saturday",  5},
72                                                  {"Sunday",    6}};
73     bool operator<(const lesson &other)const {
74         if(dayMap[this->getWeekday()]<dayMap[other.getWeekday()]){
75             return true;
76         }else if(dayMap[this->getWeekday()]==dayMap[other.getWeekday()]){
77             if(this->getStartTime()<other.getStartTime()){
78                 return true;
79             }
80             return false;
81
82         }
83         return false;
84
85     }
86
87 private:
88
89     std::string studentGroup;
90     std::string UcCode;
91     std::string weekday;
92     lessontime startTime;
93     lessontime duration;
94     lessontime endTime;
95     std::string type;
96 };
97
98
99 #endif //PROJAED_LESSON_H
```

6.7 src/lessontime.h File Reference

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

Classes

- class lessontime

  Class used to represent time.

6.8 lessontime.h

Go to the documentation of this file.

```
1
2 #ifndef PROJAED_LESSONTIME_H
3 #define PROJAED_LESSONTIME_H
4
5
6 #include <string>
7 #include <iostream>
8 #include <iomanip>
9 #include <string>
10
11 using namespace std;
15 class lessontime {
16 public:
21     explicit lessontime(double time);
22
26     lessontime();
27
33     lessontime(int hour, int minutes);
34
39     string displayHourFormat() const;
40
45     int getHour() const;
46
51     int getMinute() const;
52
53     friend std::ostream &operator«(std::ostream &os, const lessontime &t);
54
55     bool operator<(const lessontime &other)const {
56         // Compare two lessontime objects based on their hours and minutes
57         if (hour < other.hour) {
58             return true;
59         } else if (hour == other.hour && minute < other.minute) {
60             return true;
61         }
62
63         return false;
64     }
65
66     bool operator==(const lessontime &other)const {
67         // Compare two lessontime objects based on their hours and minutes
68         if (hour == other.getHour() && minute == other.getMinute()) {
69             return true;
70         }
71
72         return false;
73     }
74
75     bool operator<=(const lessontime &other)const {
76         // Compare two lessontime objects based on their hours and minutes
77         return (hour < other.hour) || (hour == other.hour && minute <= other.minute);
78     }
79
80 private:
81     int hour;
82     int minute;
83 };
84
85
86 #endif //PROJAED_LESSONTIME_H
```

6.9 src/Menu.h File Reference

```
#include "ControlUnit.h"
```

Classes

- class Menu

  Class used to represent the menu the user uses to navigate.

6.10 Menu.h

Go to the documentation of this file.

```
1
2 #ifndef PROJAED_MENU_H
3 #define PROJAED_MENU_H
4
5 #include "ControlUnit.h"
6
10 class Menu {
11 public:
15     void createMenu();
16
20     void SeeStudentSchedule();
21
25     void SeeClassSchedule();
26
30     void SeeNumStudentsInExactNUCs();
31
35     void SeeNumStudentsAtLeastNUCs();
36
40     void SeeNumStudentsAtMostNUCs();
41
45     void SeeUcFromMostStudents();
46
50     void SeeNumStudentsInNUCs();
51
55     void listingMenu();
56
60     void requestMenu();
61
65     void scheduleMenu();
66
70     void studentMenu();
71
76     void SeeStudentsInUc(function<bool(Student, Student)> comp);
77
82     void SeeStudentsInYear(function<bool(Student, Student)> comp);
83
88     void SeeStudentsInClass(function<bool(Student, Student)> comp);
89
93     void createRequest();
94
99     function<bool(Student, Student)> optionStudentMenu();
100
101 private:
102     ControlUnit Control;
103
104 };
105
106
107 #endif //PROJAED_MENU_H
```

6.11 src/RemoveRequest.h File Reference

```
#include <string>
#include "Request.h"
```

Classes

- class RemoveRequest

  Request of type remove.

6.12 RemoveRequest.h

Go to the documentation of this file.

```
1
2 #ifndef PROJAED_REMOVEREQUEST_H
3 #define PROJAED_REMOVEREQUEST_H
4
5 #include <string>
6 #include "Request.h"
7
11 class RemoveRequest :  public Request {
12 private:
13     std::string studentID;
14     std::string ucCode;
15     std::string classCode;
16
17
18 public:
25     RemoveRequest(const std::string &studentID, const std::string &ucCode,
26                   const std::string &classCode);
27
32     std::string getStudentID() const;
33
38     std::string getUCCode() const;
39
44     std::string getClassCode() const;
45
50     void setStudentID(const std::string &studentID);
51
56     void setUCCode(const std::string &ucCode);
57
62     void setClassCode(const std::string &classCode);
63 };
64
65 #endif //PROJAED_REMOVEREQUEST_H
```

6.13 src/Request.h File Reference

```
#include "student.h"
#include "studentGroup.h"
```

Classes

- class Request

  Class used to represent a generic request.

6.14 Request.h

Go to the documentation of this file.

```
1
2 #ifndef PROJAED_REQUEST_H
3 #define PROJAED_REQUEST_H
4
5 #include "student.h"
6 #include "studentGroup.h"
```

```
7
11 class Request {
12 private:
13     static int count;  // Static variable for request counter.
14     int requestId; // ID for each request.
15     bool processed;
16     std::string type;
17
18 public:
22     void static resetCount() {
23         count = 0;
24     }
25
30     Request(std::string type) {
31         count++;
32         requestId = count;
33         cout « "request id is " « requestId « " and count is " « count « endl;
34         processed = false;
35         this->type = type;
36     }
37
38     void setProcessed(bool processed) {
39         this->processed = processed;
40     }
41
46     std::string getType()const { return type; }
47
48     // Virtual function for allowing downcasting.
49     virtual void dummy() {}
50
51     virtual ~Request() {};
52 };
53
54 #endif // PROJAED_REQUEST_H
```

6.15 src/Schedule.h File Reference

```
#include <vector>
#include "lesson.h"
#include <map>
```

Classes

- class Schedule

  Class used to display a schedule.

6.16 Schedule.h

Go to the documentation of this file.

```
1
2 #ifndef PROJAED_SCHEDULE_H
3 #define PROJAED_SCHEDULE_H
4
5 #include <vector>
6 #include "lesson.h"
7 #include <map>
8
9 using namespace std;
10
14 class Schedule {
15 private:
16     vector<lesson> lessons; //the lessons that go into the schedule
17     map<pair<int, int>, string> ScheduleMap; // a schedule is made up of 30 by 6 blocks
18 public:
23     Schedule(vector<lesson> lessons);
24
28     void display();
29
30 };
31
32
33 #endif //PROJAED_SCHEDULE_H
```

6.17 src/student.h File Reference

```
#include <set>
#include <tuple>
#include <string>
#include "studentGroup.h"
```

Classes

- class Student

  Class used to represent a student.

6.18 student.h

Go to the documentation of this file.

```
1
2 #ifndef PROJAED_STUDENT_H
3 #define PROJAED_STUDENT_H
4
5 #include <set>
6 #include <tuple>
7 #include <string>
8 #include "studentGroup.h"
9
10 using namespace std;
11
15 class Student {
16 public:
20     Student() = default;
21
28     Student(string studentId, string name, set<studentGroup> group);
29
34     std::string getStudentID() const;
35
40     set<studentGroup> getStudentGroups() const;
41
46     std::string getName() const;
47
52     void setName(const std::string &newName);
53
58     void setStudentID(const std::string &studentId);
59
64     void addStudentGroup(const studentGroup& GroupToAdd);
65
70     void removeGroup(const studentGroup& GroupToRemove);
71
77     bool isInUC(const string& uc) const;
78
86     bool isInClass(const string& ucCode, const string& studentGroup) const;
87
88     bool operator<(const Student &other)const {
89         return studentID < other.studentID;
90     }
91
92     bool operator==(const Student &other)const {
93         return (this->studentID == other.studentID) && (this->name == other.name);
94     }
95
96     friend std::ostream &operator«(std::ostream &os, const Student &student);
97
98 private:
99     std::string studentID;
100     std::string name;
101     std::set<studentGroup> StudentGroups;
102 };
103
104
105 #endif //PROJAED_STUDENT_H
```

6.19 src/studentGroup.h File Reference

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

Classes

- class studentGroup

  Class used to represent a class (group of students).

6.20 studentGroup.h

Go to the documentation of this file.

```
1
2 #ifndef STUDENTGROUP_H
3 #define STUDENTGROUP_H
4
5 #include <iostream>
6 #include <string>
7
11 class studentGroup {
12 public:
16     studentGroup() = default;
17
23     studentGroup(const std::string &ucCode, const std::string &classCode);
24
29     const std::string &getClassCode()const {
30         return classCode;
31     }
32
37     const std::string &getUcCode()const {
38         return UcCode;
39     }
40
41     bool operator<(const studentGroup &other)const {
42         // Define a comparison logic here based on your criteria.
43         // For example, you can compare based on class code or other fields.
44         return this->classCode + this->UcCode < other.classCode + other.UcCode;
45     }
46
47     friend std::ostream &operator«(std::ostream &os, const studentGroup &group) {
48         os « "UcCode:  " « group.UcCode « ", Class Code:  " « group.classCode;
49         return os;
50     }
51
52
53 private:
54
55     std::string classCode;
56     std::string UcCode;
57
58
59 };
60
61 #endif
```

6.21 src/SwitchRequest.h File Reference

```
#include "Request.h"
#include <string>
```

Classes

- class SwitchRequest

  Request of type switch.

6.22 SwitchRequest.h

Go to the documentation of this file.

```
1
2 #ifndef PROJAED_SWITCHREQUEST_H
3 #define PROJAED_SWITCHREQUEST_H
4
5 #include "Request.h"
6 #include <string>
7
11 class SwitchRequest :  public Request {
12 private:
13     std::string studentID;
14     std::string ucCode1;
15     std::string ucCode2;
16     std::string classCode1;
17     std::string classCode2;
18
19 public:
28     SwitchRequest(const std::string &studentID, const std::string &ucCode1, const std::string &ucCode2,
29                   const std::string &classCode1, const std::string &classCode2);
30
35     std::string getStudentID() const;
36
41     std::string getUCCode1() const;
42
47     std::string getUCCode2() const;
48
53     std::string getClassCode1() const;
54
59     std::string getClassCode2() const;
60
61 };
62
63 #endif // PROJAED_SWITCHREQUEST_H
```

Index