

SMBC

Generated by Doxygen 1.6.1

Thu Dec 2 23:55:54 2010

Contents

1	Class 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	Course Class Reference	7
4.1.1	Member Function Documentation	8
4.1.1.1	clone	8
4.1.1.2	getDepartment	8
4.1.1.3	getDescription	8
4.1.1.4	getGler	9
4.1.1.5	getNumber	9
4.1.1.6	getPrerequisites	9
4.1.1.7	getTitle	9
4.1.1.8	hasTaken	9
4.1.1.9	read	9
4.2	CourseAlreadyExists Class Reference	11
4.2.1	Constructor & Destructor Documentation	11
4.2.1.1	CourseAlreadyExists	11

4.3	CourseList Class Reference	12
4.3.1	Detailed Description	12
4.3.2	Member Function Documentation	12
4.3.2.1	destroy	12
4.3.2.2	find	13
4.3.2.3	read	13
4.4	CourseListTest Class Reference	14
4.4.1	Member Function Documentation	14
4.4.1.1	setUp	14
4.4.1.2	tearDown	14
4.5	CourseNotFound Class Reference	15
4.5.1	Constructor & Destructor Documentation	15
4.5.1.1	CourseNotFound	15
4.6	CourseTest Class Reference	16
4.6.1	Member Function Documentation	16
4.6.1.1	setUp	16
4.6.1.2	tearDown	16
4.6.1.3	testCourseRead	16
4.7	FortyCoursesObjective Class Reference	18
4.7.1	Detailed Description	18
4.7.2	Member Function Documentation	18
4.7.2.1	clone	18
4.7.2.2	read	19
4.7.2.3	status	19
4.8	FortyCoursesObjectiveTest Class Reference	20
4.8.1	Member Function Documentation	20
4.8.1.1	setUp	20
4.8.1.2	tearDown	20
4.9	GLERObjective Class Reference	21
4.9.1	Detailed Description	21
4.9.2	Member Function Documentation	21

4.9.2.1	clone	21
4.9.2.2	read	22
4.10	GLERObjectiveTest Class Reference	23
4.10.1	Member Function Documentation	23
4.10.1.1	setUp	23
4.10.1.2	tearDown	23
4.11	GLERTest Class Reference	24
4.11.1	Member Function Documentation	24
4.11.1.1	setUp	24
4.11.1.2	tearDown	24
4.12	InvalidFile Class Reference	25
4.12.1	Constructor & Destructor Documentation	25
4.12.1.1	InvalidFile	25
4.13	InvalidGLER Class Reference	26
4.13.1	Constructor & Destructor Documentation	26
4.13.1.1	InvalidGLER	26
4.14	InvalidObjective Class Reference	27
4.14.1	Constructor & Destructor Documentation	27
4.14.1.1	InvalidObjective	27
4.15	InvalidPlan Class Reference	28
4.15.1	Constructor & Destructor Documentation	28
4.15.1.1	InvalidPlan	28
4.16	Major Class Reference	29
4.16.1	Member Function Documentation	29
4.16.1.1	copy	29
4.16.1.2	read	29
4.17	MajorAlreadyExists Class Reference	31
4.17.1	Constructor & Destructor Documentation	31
4.17.1.1	MajorAlreadyExists	31
4.18	MajorList Class Reference	32
4.18.1	Detailed Description	32

4.18.2	Member Function Documentation	32
4.18.2.1	destroy	32
4.18.2.2	find	33
4.18.2.3	getInstance	33
4.18.2.4	read	33
4.18.2.5	write	33
4.19	MajorListTest Class Reference	34
4.19.1	Member Function Documentation	34
4.19.1.1	setUp	34
4.19.1.2	tearDown	34
4.20	MajorTest Class Reference	35
4.20.1	Member Function Documentation	35
4.20.1.1	setUp	35
4.20.1.2	tearDown	35
4.21	MinimumObjective Class Reference	36
4.21.1	Detailed Description	36
4.21.2	Member Function Documentation	36
4.21.2.1	clone	36
4.21.2.2	read	37
4.22	MinimumObjectiveTest Class Reference	38
4.22.1	Member Function Documentation	38
4.22.1.1	setUp	38
4.22.1.2	tearDown	38
4.23	Objective Class Reference	39
4.23.1	Member Function Documentation	39
4.23.1.1	clone	39
4.23.1.2	read	40
4.24	OneOfObjective Class Reference	41
4.24.1	Detailed Description	41
4.24.2	Member Function Documentation	42
4.24.2.1	read	42

4.25	OneOfObjectiveTest Class Reference	43
4.25.1	Member Function Documentation	43
4.25.1.1	setUp	43
4.25.1.2	tearDown	43
4.26	Plan Class Reference	44
4.26.1	Member Function Documentation	45
4.26.1.1	canTake	45
4.26.1.2	status	45
4.27	PlanTest Class Reference	46
4.27.1	Member Function Documentation	46
4.27.1.1	initCourseVectors	46
4.27.1.2	setUp	46
4.27.1.3	tearDown	47
4.28	SingleObjective Class Reference	48
4.28.1	Detailed Description	48
4.28.2	Member Function Documentation	49
4.28.2.1	read	49
4.29	SingleObjectiveTest Class Reference	50
4.29.1	Member Function Documentation	50
4.29.1.1	setUp	50
4.29.1.2	tearDown	50
4.30	Student Class Reference	51
4.31	StudentNotFound Class Reference	52
4.31.1	Constructor & Destructor Documentation	52
4.31.1.1	StudentNotFound	52
4.32	StudentTest Class Reference	53
4.32.1	Member Function Documentation	54
4.32.1.1	setUp	54
4.32.1.2	tearDown	54

5.1	Course.cc File Reference	55
5.1.1	Detailed Description	55
5.2	Course.h File Reference	56
5.2.1	Detailed Description	56
5.3	CourseList.cc File Reference	58
5.3.1	Detailed Description	58
5.4	CourseList.h File Reference	59
5.4.1	Detailed Description	59
5.5	CourseListTest.cc File Reference	60
5.5.1	Detailed Description	60
5.6	CourseListTest.h File Reference	61
5.6.1	Detailed Description	61
5.7	CourseTest.cc File Reference	62
5.7.1	Detailed Description	62
5.8	CourseTest.h File Reference	63
5.8.1	Detailed Description	63
5.9	Exceptions.h File Reference	64
5.9.1	Detailed Description	64
5.10	FortyCoursesObjective.cc File Reference	65
5.10.1	Detailed Description	65
5.11	FortyCoursesObjective.h File Reference	66
5.11.1	Detailed Description	66
5.12	FortyCoursesObjectiveTest.cc File Reference	67
5.12.1	Detailed Description	67
5.13	FortyCoursesObjectiveTest.h File Reference	68
5.13.1	Detailed Description	68
5.14	GLER.h File Reference	69
5.14.1	Detailed Description	69
5.15	GLERObjective.cc File Reference	70
5.15.1	Detailed Description	70
5.16	GLERObjective.h File Reference	71

5.16.1 Detailed Description	71
5.17 GLERObjectiveTest.cc File Reference	72
5.17.1 Detailed Description	72
5.18 GLERObjectiveTest.h File Reference	73
5.18.1 Detailed Description	73
5.19 GLERTest.cc File Reference	74
5.19.1 Detailed Description	74
5.20 GLERTest.h File Reference	75
5.20.1 Detailed Description	75
5.21 Major.cc File Reference	76
5.21.1 Detailed Description	76
5.22 Major.h File Reference	77
5.22.1 Detailed Description	77
5.23 MajorList.cc File Reference	78
5.23.1 Detailed Description	78
5.23.2 Function Documentation	78
5.23.2.1 operator<<	78
5.23.2.2 operator>>	78
5.24 MajorList.h File Reference	79
5.24.1 Detailed Description	79
5.25 MajorListTest.cc File Reference	80
5.25.1 Detailed Description	80
5.26 MajorListTest.h File Reference	81
5.26.1 Detailed Description	81
5.27 MajorTest.cc File Reference	82
5.27.1 Detailed Description	82
5.28 MajorTest.h File Reference	83
5.28.1 Detailed Description	83
5.29 MinimumObjective.cc File Reference	84
5.29.1 Detailed Description	84
5.30 MinimumObjective.h File Reference	85

5.30.1 Detailed Description	85
5.31 MinimumObjectiveTest.cc File Reference	86
5.31.1 Detailed Description	86
5.32 MinimumObjectiveTest.h File Reference	87
5.32.1 Detailed Description	87
5.33 Objective.h File Reference	88
5.33.1 Detailed Description	88
5.34 OneOfObjective.cc File Reference	89
5.34.1 Detailed Description	89
5.35 OneOfObjective.h File Reference	90
5.35.1 Detailed Description	90
5.36 OneOfObjectiveTest.cc File Reference	91
5.36.1 Detailed Description	91
5.37 OneOfObjectiveTest.h File Reference	92
5.37.1 Detailed Description	92
5.38 Plan.cc File Reference	93
5.38.1 Detailed Description	93
5.39 Plan.h File Reference	94
5.39.1 Detailed Description	94
5.40 PlanTest.cc File Reference	95
5.40.1 Detailed Description	95
5.41 PlanTest.h File Reference	96
5.41.1 Detailed Description	96
5.42 SingleObjective.cc File Reference	97
5.42.1 Detailed Description	97
5.43 SingleObjective.h File Reference	98
5.43.1 Detailed Description	98
5.44 SingleObjectiveTest.cc File Reference	99
5.44.1 Detailed Description	99
5.45 SingleObjectiveTest.h File Reference	100
5.45.1 Detailed Description	100

5.46 Student.cc File Reference	101
5.46.1 Detailed Description	101
5.47 Student.h File Reference	102
5.47.1 Detailed Description	102
5.48 StudentTest.cc File Reference	103
5.48.1 Detailed Description	103
5.49 StudentTest.h File Reference	104
5.49.1 Detailed Description	104
5.50 tester.cc File Reference	105
5.50.1 Detailed Description	105

Chapter 1

Class Index

1.1 Class Hierarchy

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

Course	7
CourseAlreadyExists	11
CourseList	12
CourseListTest	14
CourseNotFound	15
CourseTest	16
FortyCoursesObjectiveTest	20
GLERObjectiveTest	23
GLERTest	24
InvalidFile	25
InvalidGLER	26
InvalidObjective	27
InvalidPlan	28
Major	29
MajorAlreadyExists	31
MajorList	32
MajorListTest	34
MajorTest	35
MinimumObjectiveTest	38
Objective	39
FortyCoursesObjective	18
GLERObjective	21
MinimumObjective	36
OneOfObjective	41
SingleObjective	48

OneOfObjectiveTest	43
Plan	44
PlanTest	46
SingleObjectiveTest	50
Student	51
StudentNotFound	52
StudentTest	53

Chapter 2

Class Index

2.1 Class List

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

Course (A class to represent a University of Lethbridge course)	7
CourseAlreadyExists (Exception thrown when a duplicate course is being read into a CourseList)	11
CourseList (A master list of courses)	12
CourseListTest (Class to test functionality of the CourseList class)	14
CourseNotFound (Exception class to handle invalid course names)	15
CourseTest (Class to test functionality of the Course class)	16
FortyCoursesObjective (An objective requiring 40 courses to be taken) . . .	18
FortyCoursesObjectiveTest (Class to test functionality of the FortyCoursesObjective class)	20
GLERObjective (An objective requiring 4 of a particular GLER category to be taken)	21
GLERObjectiveTest (Class to test functionality of the GLERObjective class)	23
GLERTest (Class to test functionality of the GLER class)	24
InvalidFile (Exception class to handle invalid file names)	25
InvalidGLER (Exception thrown when an invalid GLER is encountered) . . .	26
InvalidObjective (Exception class to handle invalid Objectives)	27
InvalidPlan (Exception thrown when an invalid Plan is created or encountered)	28
Major (A class to represent a University of Lethbridge major)	29
MajorAlreadyExists (Exception thrown when a duplicate major is being read into a MajorList)	31
MajorList (A master list of majors)	32
MajorListTest (Class to test functionality of the MajorList class)	34
MajorTest (Class to test functionality of the Major class)	35

MinimumObjective (An Objective requiring a given number of courses at a particular level to be completed)	36
MinimumObjectiveTest (Class to test functionality of the MinimumObjective class)	38
Objective (An abstract base class to define the requirements of a major and track their completion status)	39
OneOfObjective (An objective requiring a single Course out of a list of Courses to be completed)	41
OneOfObjectiveTest (Class to test functionality of the OneOfObjective class)	43
Plan (Builds a potential future plan for a Student)	44
PlanTest (Class to test functionality of the Plan class)	46
SingleObjective (An objective requiring one, specific course to be completed)	48
SingleObjectiveTest (Class to test functionality of the SingleObjective class)	50
Student (A class to represent a student's course status)	51
StudentNotFound (Exception class to handle invalid student names)	52
StudentTest (Class to test functionality of the Student class)	53

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Course.cc (A class to represent a University of Lethbridge course)	55
Course.h (A class to represent a University of Lethbridge course)	56
CourseList.cc (A master list of courses)	58
CourseList.h (A master list of courses)	59
CourseListTest.cc (Implementation of the unit tests for the CourseList class)	60
CourseListTest.h (Unit tests for the CourseList class)	61
CourseTest.cc (Tests the member functions of Course to ensure proper func- tionality)	62
CourseTest.h	63
Exceptions.h (Exception library for the application)	64
FortyCoursesObjective.cc (The main objective for completing a generic ma- jor)	65
FortyCoursesObjective.h (The main objective for completing a generic major)	66
FortyCoursesObjectiveTest.cc	67
FortyCoursesObjectiveTest.h	68
GLER.h (An enumerated data type, either SCIENCE, SOCIAL_SCIENCE, or FINE_ARTS_HUMANITY)	69
GLERObjective.cc (An objective requiring 4 of a particular GLER category to be taken)	70
GLERObjective.h (An objective requiring 4 of a particular GLER category to be taken)	71
GLERObjectiveTest.cc	72
GLERObjectiveTest.h	73
GLERTest.cc	74
GLERTest.h	75

Major.cc (A class to represent a University of Lethbridge major)	76
Major.h (A class to represent a University of Lethbridge major)	77
MajorList.cc (A master list of majors)	78
MajorList.h (A master list of majors)	79
MajorListTest.cc	80
MajorListTest.h	81
MajorTest.cc	82
MajorTest.h	83
MinimumObjective.cc (An Objective requiring a given number of courses at a particular level to be completed)	84
MinimumObjective.h (An Objective requiring a given number of courses at a particular level to be completed)	85
MinimumObjectiveTest.cc	86
MinimumObjectiveTest.h	87
Objective.h (An abstract base class to define the requirements of a major and track their completion status)	88
OneOfObjective.cc (An objective requiring a single Course out of a list of Courses to be completed)	89
OneOfObjective.h (An objective requiring a single Course out of a list of Courses to be completed)	90
OneOfObjectiveTest.cc	91
OneOfObjectiveTest.h	92
Plan.cc (Builds a potential future plan for a Student)	93
Plan.h (Builds a potential future plan for a Student)	94
PlanTest.cc	95
PlanTest.h	96
SingleObjective.cc (An objective requiring one, specific course to be com- pleted)	97
SingleObjective.h (An objective requiring one, specific course to be com- pleted)	98
SingleObjectiveTest.cc	99
SingleObjectiveTest.h	100
Student.cc (A class to represent a student's course status)	101
Student.h (A class to represent a student's course status)	102
StudentTest.cc	103
StudentTest.h	104
tester.cc	105

Chapter 4

Class Documentation

4.1 Course Class Reference

A class to represent a University of Lethbridge course.

```
#include <Course.h>
```

Public Member Functions

- `Course ()`
Constructor for the `Course` class.
- `Course (const Course &cObj)`
Copy constructor for the `Course` class.
- `Course * clone ()`
Clone function.
- `const GLER getGler () const`
Accessor function for `GLER` category.
- `const std::string getDepartment () const`
Accessor function for `std::string` department.
- `int getNumber () const`
Accessor function for unsigned short number.
- `const std::vector< Objective * > getPrerequisites () const`

Accessor function for vector of prerequisites.

- `const std::string getTitle () const`
Accessor function for title.
- `const std::string getDescription () const`
Accessor function for description.
- `bool hasTaken (std::vector< Course * > courseVector) const`
Checks if course is already in the vector.
- `void read (std::istream &iStr) throw (InvalidFile)`
Initializes the course's members.
- `void write (std::ostream &os) const`
Prints the course's members.

4.1.1 Member Function Documentation

4.1.1.1 `Course * Course::clone ()`

Returns:

A copy of this course.

References `Course()`.

4.1.1.2 `const std::string Course::getDepartment () const`

Returns:

The course's department

Referenced by `Course()`, `hasTaken()`, and `CourseTest::testCourseAccessors()`.

4.1.1.3 `const std::string Course::getDescription () const`

Returns:

The description of the course

Referenced by `Course()`, `operator==()`, and `CourseTest::testCourseAccessors()`.

4.1.1.4 const GLER Course::getGler () const**Returns:**

The course's GLER

Referenced by Course(), operator==(), and CourseTest::testCourseAccessors().

4.1.1.5 int Course::getNumber () const**Returns:**

The course's number

Referenced by Course(), hasTaken(), operator==(), and CourseTest::testCourseAccessors().

4.1.1.6 const std::vector< Objective * > Course::getPrerequisites () const**Returns:**

The vector of prerequisites

Referenced by Student::canTake(), Course(), and operator==().

4.1.1.7 const std::string Course::getTitle () const**Returns:**

The title of the course

Referenced by Course(), operator==(), and CourseTest::testCourseAccessors().

4.1.1.8 bool Course::hasTaken (std::vector< Course * > *courseVector*) const**Returns:**

true if the course is in the vector, false otherwise.

References getDepartment(), and getNumber().

4.1.1.9 void Course::read (std::istream & *iStr*) throw (InvalidFile)

Should be in the format:

[department] [number]

[title]

[description]

[GLER category]

P [tag of objective] [prerequisite1]

P [tag of objective] [prerequisite2]

...

Referenced by operator>>(), CourseTest::testCourseAccessors(),
CourseTest::testCourseRead(), and CourseTest::testCourseWrite().

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

- [Course.h](#)
- [Course.cc](#)

4.2 CourseAlreadyExists Class Reference

Exception thrown when a duplicate course is being read into a [CourseList](#).

```
#include <Exceptions.h>
```

Inherits `std::runtime_error`.

Public Member Functions

- [CourseAlreadyExists](#) (const `std::string` &*arg*)

4.2.1 Constructor & Destructor Documentation

4.2.1.1 CourseAlreadyExists::CourseAlreadyExists (const `std::string` & *arg*) [`inline`]

Constructor

Parameters:

- arg* the message to be displayed

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

- [Exceptions.h](#)

4.3 CourseList Class Reference

A master list of courses.

```
#include <CourseList.h>
```

Public Member Functions

- void [destroy](#) ()
Destroys a [CourseList](#) when it is no longer needed.
- void [read](#) (std::istream &iStr) throw (CourseAlreadyExists, InvalidFile)
Calls [Course](#)'s read function for every [Course](#) in the file.
- void [write](#) (std::ostream &os) const
Prints only the departments and numbers of the courses.
- [Course](#) * [find](#) (const std::string &department, const int &number)
Searches for a specified [Course](#) in the vector of Courses.

Static Public Member Functions

- static [CourseList](#) * [getInstance](#) ()
Creates an instance if there isn't one or returns the only one.

4.3.1 Detailed Description

This class is a Singleton

4.3.2 Member Function Documentation

4.3.2.1 void CourseList::destroy ()

This function removes all the Courses contained in the [CourseList](#) when the application exits. It isn't even really necessary, because the only time that the [CourseList](#) should be deleted is when the program terminates, and even then, without this function the operating system would simply free the memory that was used by it anyway. So, **technically**, that is not a memory leak. I am not going to naively say that it is never a memory leak, because some OSes don't free the memory, so we'll include the [destroy\(\)](#) function anyway. -CR

Go through the [CourseList](#) and delete all Course*

4.3.2.2 Course * CourseList::find (const std::string & *department*, const int & *number*)

Returns:

A pointer to a [Course](#)

4.3.2.3 void CourseList::read (std::istream & *iStr*) throw (CourseAlreadyExists, InvalidFile)

Should be in the format:

C

[Course1]

C

[Course2]

...

Referenced by operator>>().

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

- [CourseList.h](#)
- [CourseList.cc](#)

4.4 CourseListTest Class Reference

class to test functionality of the [CourseList](#) class

```
#include <CourseListTest.h>
```

Public Member Functions

- void [setUp](#) ()
override setUp to create variable
- void [tearDown](#) ()
override tearDown to clean up
- void **testRead** ()
- void **testFind** ()
- void **testSingleton** ()

4.4.1 Member Function Documentation

4.4.1.1 void CourseListTest::setUp ()

Create some variables to test with.

References [CourseList::getInstance\(\)](#).

4.4.1.2 void CourseListTest::tearDown ()

Free space used by allocated variables.

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

- [CourseListTest.h](#)
- [CourseListTest.cc](#)

4.5 CourseNotFound Class Reference

Exception class to handle invalid course names.

```
#include <Exceptions.h>
```

Inherits `std::runtime_error`.

Public Member Functions

- [CourseNotFound](#) (const `std::string` &*arg*)
Constructor.

4.5.1 Constructor & Destructor Documentation

4.5.1.1 `CourseNotFound::CourseNotFound` (const `std::string` & *arg*) [`inline`]

Parameters:

arg the message to be displayed

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

- [Exceptions.h](#)

4.6 CourseTest Class Reference

class to test functionality of the [Course](#) class

```
#include <CourseTest.h>
```

Public Member Functions

- void [setUp](#) ()
override `setUp` to create variable
- void [tearDown](#) ()
override `tearDown` to clean up
- void [testCourseRead](#) ()
- void [testCourseWrite](#) ()
Test the `Course::write` function writes the correct data to the ostream.
- void [testCourseAccessors](#) ()
Tests all the accessor functions in `Course` to ensure they return the correct information.

4.6.1 Member Function Documentation

4.6.1.1 void CourseTest::setUp ()

Create some variables to test with.

4.6.1.2 void CourseTest::tearDown ()

Free space used by allocated variables.

4.6.1.3 void CourseTest::testCourseRead ()

Test the [Course::read](#) function to ensure that the correct exceptions are thrown when an invalid course is read

Checking courses without prerequisites are read properly

Checking courses with prerequisites are read properly; destroyed in `tearDown`

Checks that invalid data cannot be read into a course

References `Course::read()`.

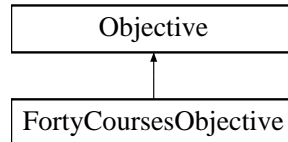
The documentation for this class was generated from the following files:

- [CourseTest.h](#)
- [CourseTest.cc](#)

4.7 FortyCoursesObjective Class Reference

An objective requiring 40 courses to be taken.

#include <FortyCoursesObjective.h> Inheritance diagram for FortyCoursesObjective::



Public Member Functions

- [FortyCoursesObjective](#) * [clone](#) () const
Return a pointer to a cpy of this object.
- bool [status](#) (std::ostream &os, const std::vector< [Course](#) * > &coursesTaken) const
Counts to see if there are at least 40 courses in coursesTaken.
- void [read](#) (std::istream &iStr)
Initializes the objective's members.
- void [write](#) (std::ostream &os) const
Writes the objective to the ostream.

4.7.1 Detailed Description

No more than 10 of which are 1000 level No more than 20 from one department A subclass of [Objective](#)

4.7.2 Member Function Documentation

4.7.2.1 [FortyCoursesObjective](#) * [FortyCoursesObjective::clone](#) () const [virtual]

Returns:

a pointer to a copy of the current [Objective](#)

Implements [Objective](#).

4.7.2.2 void FortyCoursesObjective::read (std::istream & *iStr*) [inline, virtual]

Does nothing in this case

Implements [Objective](#).

4.7.2.3 bool FortyCoursesObjective::status (std::ostream & *os*, const std::vector< Course * > & *coursesTaken*) const [virtual]

Not counting courses that exceed the 10 1000 level limit or the 20 per department limit

Implements [Objective](#).

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

- [FortyCoursesObjective.h](#)
- [FortyCoursesObjective.cc](#)

4.8 FortyCoursesObjectiveTest Class Reference

class to test functionality of the [FortyCoursesObjective](#) class

```
#include <FortyCoursesObjectiveTest.h>
```

Public Member Functions

- void [setUp](#) ()
override setUp to create variable
- void [tearDown](#) ()
override tearDown to clean up

4.8.1 Member Function Documentation

4.8.1.1 void FortyCoursesObjectiveTest::setUp ()

Create some variables to test with.

4.8.1.2 void FortyCoursesObjectiveTest::tearDown ()

Free space used by allocated variables.

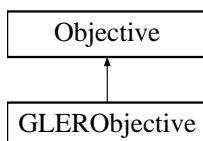
The documentation for this class was generated from the following files:

- [FortyCoursesObjectiveTest.h](#)
- [FortyCoursesObjectiveTest.cc](#)

4.9 GLERObjective Class Reference

An objective requiring 4 of a particular GLER category to be taken.

`#include <GLERObjective.h>` Inheritance diagram for GLERObjective::



Public Member Functions

- **GLERObjective** ([GLER](#) gler=SCIENCE)
- **GLERObjective * clone** () const
Return a pointer to a cpy of this object.
- bool **status** (std::ostream &os, const std::vector< [Course](#) * > &coursesTaken) const
Compares gler in coursesTaken to see if 4 match gler.
- void **read** (std::istream &iStr) throw (InvalidGLER)
Initializes the objective's members.
- void **write** (std::ostream &os) const
Writes the objective to the ostream.

4.9.1 Detailed Description

A subclass of [Objective](#)

4.9.2 Member Function Documentation

4.9.2.1 GLERObjective * GLERObjective::clone () const [virtual]

Returns:

a pointer to a copy of the current [Objective](#)

Implements [Objective](#).

4.9.2.2 void GLERObjective::read (std::istream & *iStr*) throw (InvalidGLER) [virtual]

Should be in the format:

[glerType]

Implements [Objective](#).

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

- [GLERObjective.h](#)
- [GLERObjective.cc](#)

4.10 GLERObjectiveTest Class Reference

class to test functionality of the [GLERObjective](#) class

```
#include <GLERObjectiveTest.h>
```

Public Member Functions

- void [initCourseVectors](#) ()
Creates test courses to use.
- void [setUp](#) ()
override setUp to create variable
- void [tearDown](#) ()
override tearDown to clean up
- void [testStatus](#) ()
Tests the status function for GLER.
- void [testRead](#) ()
Tests read function.
- void [testWrite](#) ()
Tests write function.

4.10.1 Member Function Documentation

4.10.1.1 void GLERObjectiveTest::setUp ()

Create some variables to test with.

References [initCourseVectors\(\)](#).

4.10.1.2 void GLERObjectiveTest::tearDown ()

Free space used by allocated variables.

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

- [GLERObjectiveTest.h](#)
- [GLERObjectiveTest.cc](#)

4.11 GLERTest Class Reference

class to test functionality of the GLER class

```
#include <GLERTest.h>
```

Public Member Functions

- void [setUp](#) ()
override setUp to create variable
- void [tearDown](#) ()
override tearDown to clean up

4.11.1 Member Function Documentation

4.11.1.1 void GLERTest::setUp ()

Create some variables to test with.

4.11.1.2 void GLERTest::tearDown ()

Free space used by allocated variables.

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

- [GLERTest.h](#)
- [GLERTest.cc](#)

4.12 InvalidFile Class Reference

Exception class to handle invalid file names.

```
#include <Exceptions.h>
```

Inherits `std::runtime_error`.

Public Member Functions

- [InvalidFile](#) (const `std::string` &*arg*)
Constructor.

4.12.1 Constructor & Destructor Documentation

4.12.1.1 InvalidFile::InvalidFile (const `std::string` &*arg*) [inline]

Parameters:

arg the message to be displayed

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

- [Exceptions.h](#)

4.13 InvalidGLER Class Reference

Exception thrown when an invalid GLER is encountered.

```
#include <Exceptions.h>
```

Inherits `std::runtime_error`.

Public Member Functions

- [InvalidGLER](#) (const std::string &arg)

4.13.1 Constructor & Destructor Documentation

4.13.1.1 InvalidGLER::InvalidGLER (const std::string & *arg*) [inline]

Constructor

Parameters:

arg the message to be displayed

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

- [Exceptions.h](#)

4.14 InvalidObjective Class Reference

Exception class to handle invalid Objectives.

```
#include <Exceptions.h>
```

Inherits `std::runtime_error`.

Public Member Functions

- [InvalidObjective](#) (const `std::string` &*arg*)

4.14.1 Constructor & Destructor Documentation

4.14.1.1 InvalidObjective::InvalidObjective (const `std::string` & *arg*) [`inline`]

Constructor

Parameters:

arg the message to be displayed

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

- [Exceptions.h](#)

4.15 InvalidPlan Class Reference

Exception thrown when an invalid [Plan](#) is created or encountered.

```
#include <Exceptions.h>
```

Inherits `std::runtime_error`.

Public Member Functions

- [InvalidPlan](#) (const `std::string` &*arg*)

4.15.1 Constructor & Destructor Documentation

4.15.1.1 InvalidPlan::InvalidPlan (const `std::string` &*arg*) [inline]

Constructor

Parameters:

arg the message to be displayed

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

- [Exceptions.h](#)

4.16 Major Class Reference

A class to represent a University of Lethbridge major.

```
#include <Major.h>
```

Public Member Functions

- void `copy` (const `Major` &orig)
Make a copy of the data in the `Major`.
- `Major` (`Major` &m)
- void `read` (std::istream &iStr)
Initializes the `Major`'s members.
- void `write` (std::ostream &os) const
Prints the `Major`'s members.
- bool `status` (std::ostream &os, const std::vector< `Course` * > &coursesTaken) const
Calls each `Objective`'s status.
- const std::string & `getTitle` () const
Get the title of the `Major`.

4.16.1 Member Function Documentation

4.16.1.1 void Major::copy (const Major & orig)

Parameters:

orig imports the `Major` to make a copy of

4.16.1.2 void Major::read (std::istream & iStr)

Should be in the format:

[title]

[objective1]

[objective2]

...

Referenced by operator>>().

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

- [Major.h](#)
- [Major.cc](#)

4.17 MajorAlreadyExists Class Reference

Exception thrown when a duplicate major is being read into a [MajorList](#).

```
#include <Exceptions.h>
```

Inherits `std::runtime_error`.

Public Member Functions

- [MajorAlreadyExists](#) (const `std::string` &*arg*)

4.17.1 Constructor & Destructor Documentation

4.17.1.1 MajorAlreadyExists::MajorAlreadyExists (const `std::string` & *arg*) [`inline`]

Constructor

Parameters:

- *arg* the message to be displayed

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

- [Exceptions.h](#)

4.18 MajorList Class Reference

A master list of majors.

```
#include <MajorList.h>
```

Public Member Functions

- void [destroy](#) ()
Destroys the singleton [MajorList](#) in memory.
- void [read](#) (std::istream &iStr) throw (MajorAlreadyExists, InvalidFile)
Calls Major's read function for every major in the file.
- void [write](#) (std::ostream &os) const
Prints only the names of the majors.
- [Major](#) * [find](#) (const std::string &title)
Returns a pointer to a specified [Major](#) in majors.

Static Public Member Functions

- static [MajorList](#) * [getInstance](#) ()
Creates an instance if there isn't one or returns the only one.

4.18.1 Detailed Description

This class is a Singleton

4.18.2 Member Function Documentation

4.18.2.1 void MajorList::destroy ()

This function removes all the Majors contained in the [MajorList](#) when the application exits. It isn't even really necessary, because the only time that the [MajorList](#) should be deleted is when the program terminates, and even then, without this function the operating system would simply free the memory that was used by it anyway. So, **technically**, that is not a memory leak. I am not going to naively say that it is never a memory leak, because some OSes don't free the memory, so we'll include the [destroy\(\)](#) function anyway. -CR

4.18.2.2 Major * MajorList::find (const std::string & *title*)

Finds the course in the major list matching title.

4.18.2.3 MajorList * MajorList::getInstance () [static]

Gets the instance of the [Major](#) List.

Returns:

a pointer to the [Major](#) List Singleton

4.18.2.4 void MajorList::read (std::istream & *iStr*) throw (MajorAlreadyExists, InvalidFile)

Reads the data into the [Major](#) List.

Should be in the format:

M

[major1]

M

[major2]

...

Each [Major](#) starts with the 'M' character, followed by the format used for the [Major](#) class.

Referenced by operator>>().

4.18.2.5 void MajorList::write (std::ostream & *os*) const

Writes the major list to the ostream.

Referenced by operator<<().

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

- [MajorList.h](#)
- [MajorList.cc](#)

4.19 MajorListTest Class Reference

class to test functionality of the [MajorList](#) class

```
#include <MajorListTest.h>
```

Public Member Functions

- void [setUp](#) ()
override setUp to create variable
- void [tearDown](#) ()
override tearDown to clean up

4.19.1 Member Function Documentation

4.19.1.1 void MajorListTest::setUp ()

Create some variables to test with.

4.19.1.2 void MajorListTest::tearDown ()

Free space used by allocated variables.

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

- [MajorListTest.h](#)
- [MajorListTest.cc](#)

4.20 MajorTest Class Reference

class to test functionality of the [Major](#) class

```
#include <MajorTest.h>
```

Public Member Functions

- void [setUp](#) ()
override setUp to create variable
- void [tearDown](#) ()
override tearDown to clean up

4.20.1 Member Function Documentation

4.20.1.1 void MajorTest::setUp ()

Create some variables to test with.

4.20.1.2 void MajorTest::tearDown ()

Free space used by allocated variables.

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

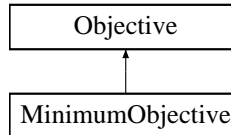
- [MajorTest.h](#)
- [MajorTest.cc](#)

4.21 MinimumObjective Class Reference

An [Objective](#) requiring a given number of courses at a particular level to be completed.

```
#include <MinimumObjective.h>
```

Inheritance diagram for MinimumObjective::



Public Member Functions

- `bool status (std::ostream &os, const std::vector< Course * > &coursesTaken) const`
Compares `coursesTaken` until `n` match the level of the objective.
- `void read (std::istream &iStr)`
Initializes the objective's members.
- `void write (std::ostream &os) const`
Writes the [Objective](#) to the ostream.
- `MinimumObjective * clone () const`
Return a pointer to a copy of this object.

4.21.1 Detailed Description

This is a subclass of [Objective](#)

4.21.2 Member Function Documentation

4.21.2.1 `MinimumObjective * MinimumObjective::clone () const` `[virtual]`

Returns:

a pointer to a copy of the current [Objective](#)

Implements [Objective](#).

4.21.2.2 void MinimumObjective::read (std::istream & *iStr*) [virtual]

Should be in the format:

[minCourses] [department] [level]

Implements [Objective](#).

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

- [MinimumObjective.h](#)
- [MinimumObjective.cc](#)

4.22 MinimumObjectiveTest Class Reference

class to test functionality of the [MinimumObjective](#) class

```
#include <MinimumObjectiveTest.h>
```

Public Member Functions

- void [setUp](#) ()
override setUp to create variable
- void [tearDown](#) ()
override tearDown to clean up

4.22.1 Member Function Documentation

4.22.1.1 void MinimumObjectiveTest::setUp ()

Create some variables to test with.

4.22.1.2 void MinimumObjectiveTest::tearDown ()

Free space used by allocated variables.

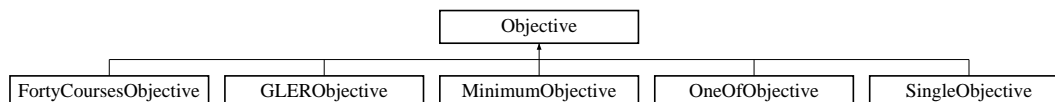
The documentation for this class was generated from the following files:

- [MinimumObjectiveTest.h](#)
- [MinimumObjectiveTest.cc](#)

4.23 Objective Class Reference

An abstract base class to define the requirements of a major and track their completion status.

`#include <Objective.h>` Inheritance diagram for Objective::



Public Member Functions

- virtual **Objective** * **clone** () const =0
Return a pointer to a copy of this object.
- virtual bool **status** (std::ostream &os, const std::vector< **Course** * > &courseList) const =0
Determines the objective's completion.
- virtual bool **isEqual** (**Objective** *o)
- virtual void **read** (std::istream &iStr)=0
Initializes the objective's members.
- virtual void **write** (std::ostream &os) const =0
Writes the objective to the ostream.
- virtual std::istream & **operator>>** (std::istream &is)
Overloads the input stream operator for all Objectives.
- virtual std::ostream & **operator<<** (std::ostream &os) const
Overloads the output stream operator for all Objectives.

4.23.1 Member Function Documentation

4.23.1.1 virtual Objective* Objective::clone () const [pure virtual]

Returns:

a pointer to a copy of the current **Objective**

Implemented in [FortyCoursesObjective](#), [GLERObjective](#), [MinimumObjective](#), [OneOfObjective](#), and [SingleObjective](#).

4.23.1.2 virtual void Objective::read (std::istream & *iStr*) [pure virtual]

Should be in the format:

[type] [[Objective](#)]

where [type] is:

G for [GLERObjective](#)

M for [MinimumObjective](#)

O for [OneOfObjective](#)

S for [SingleObjective](#)

Implemented in [FortyCoursesObjective](#), [GLERObjective](#), [MinimumObjective](#), [OneOfObjective](#), and [SingleObjective](#).

Referenced by operator>>().

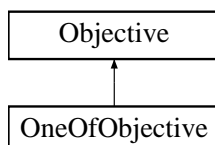
The documentation for this class was generated from the following file:

- [Objective.h](#)

4.24 OneOfObjective Class Reference

An objective requiring a single [Course](#) out of a list of Courses to be completed.

`#include <OneOfObjective.h>`Inheritance diagram for OneOfObjective::



Public Member Functions

- **OneOfObjective** (const [OneOfObjective](#) &oObj)
- [OneOfObjective](#) * [clone](#) () const
Return of a copy of the [Objective](#).
- bool [status](#) (std::ostream &os, const std::vector< [Course](#) * > &coursesTaken) const
Compares coursesTaken to oneOf until one match is found.
- const bool [isEqual](#) ([OneOfObjective](#) *oObj) const
equality test between two OneOfObjectives
- std::vector< std::string > [getDeptVector](#) () const
Returns the vector of departments.
- std::vector< int > [getNumberVector](#) () const
Returns the vector of course numbers.
- void [read](#) (std::istream &iStr)
Initializes the objective's members.
- void [write](#) (std::ostream &os) const
Writes the objective to the ostream.

4.24.1 Detailed Description

This is a subclass of [Objective](#)

4.24.2 Member Function Documentation

4.24.2.1 `void OneOfObjective::read (std::istream & iStr) [virtual]`

Should be in the format:

[dept1] [number1] [dept2] [number2] ...

Implements [Objective](#).

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

- [OneOfObjective.h](#)
- [OneOfObjective.cc](#)

4.25 OneOfObjectiveTest Class Reference

class to test functionality of the [OneOfObjective](#) class

```
#include <OneOfObjectiveTest.h>
```

Public Member Functions

- void [setUp](#) ()
override setUp to create variable
- void [tearDown](#) ()
override tearDown to clean up

4.25.1 Member Function Documentation

4.25.1.1 void OneOfObjectiveTest::setUp ()

Create some variables to test with.

4.25.1.2 void OneOfObjectiveTest::tearDown ()

Free space used by allocated variables.

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

- [OneOfObjectiveTest.h](#)
- [OneOfObjectiveTest.cc](#)

4.26 Plan Class Reference

Builds a potential future plan for a [Student](#).

```
#include <Plan.h>
```

Public Member Functions

- **Plan** ([Student](#) *s=NULL)
- [Major](#) * [getMajor](#) () const
Returns a pointer to the plan's major.
- std::vector< [Course](#) * > [getCourses](#) () const
Returns the vector of course pointers.
- void [write](#) (std::ostream &os) const throw (InvalidPlan)
Prints the plan's members.
- void [changeMajor](#) ([Major](#) *newMajor)
Changes the plan's major.
- void [addCourse](#) ([Course](#) *newCourse) throw (CourseAlreadyExists)
Adds a pointer to a course in cl to coursesToTake.
- void [removeCourse](#) (const [Course](#) *oldCourse)
Removes the pointer to c from coursesToTake.
- bool [status](#) (std::ostream &os) const throw (InvalidPlan)
Calls major's status function with coursesToTake and coursesTaken.
- bool [canTake](#) (const [Course](#) *potentialCourse) const throw (InvalidPlan)
Calls c's Objectives' statuses with coursesToTake and coursesTaken.
- void [commitChanges](#) () throw (InvalidPlan)
Updates the student's coursesTaken with the plan's coursesToTake.
- std::vector< [Course](#) * > [combineVectors](#) () const throw (InvalidPlan)
Combine the coursesTaken vector from student and coursesToTake.

4.26.1 Member Function Documentation

4.26.1.1 `bool Plan::canTake (const Course * potentialCourse) const throw (InvalidPlan)`

Uses student's major if one is not defined

4.26.1.2 `bool Plan::status (std::ostream & os) const throw (InvalidPlan)`

Uses student's major if one is not defined

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

- [Plan.h](#)
- [Plan.cc](#)

4.27 PlanTest Class Reference

class to test functionality of the [Plan](#) class

```
#include <PlanTest.h>
```

Public Member Functions

- void [setUp](#) ()
override setUp to create variable
- void [tearDown](#) ()
override tearDown to clean up
- void [initCourseVectors](#) ()
- void [testAddCourse](#) ()
- void [testCanTake](#) ()
- void [testChangeMajor](#) ()
- void [testCombineVectors](#) ()
- void [testCommitChanges](#) ()
- void [testGetCourses](#) ()
- void [testGetMajor](#) ()
- void [testRemoveCourse](#) ()
- void [testStatus](#) ()
- void [testWrite](#) ()

4.27.1 Member Function Documentation

4.27.1.1 void PlanTest::initCourseVectors ()

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

Referenced by [setUp](#)().

4.27.1.2 void PlanTest::setUp ()

Create some variables to test with.

References [initCourseVectors](#)().

4.27.1.3 void PlanTest::tearDown ()

Free space used by allocated variables.

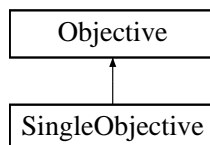
The documentation for this class was generated from the following files:

- [PlanTest.h](#)
- [PlanTest.cc](#)

4.28 SingleObjective Class Reference

An objective requiring one, specific course to be completed.

#include <SingleObjective.h> Inheritance diagram for SingleObjective::



Public Member Functions

- **SingleObjective** (const [SingleObjective](#) &sObj)
- [SingleObjective](#) * **clone** () const
Return a copy of the [Objective](#).
- bool [status](#) (std::ostream &os, const std::vector< [Course](#) * > &coursesTaken) const
Searches for course in coursesTaken.
- const bool [isEqual](#) ([SingleObjective](#) *sObj) const
equality test between two SingleObjectives
- std::string [getDept](#) () const
Returns the department.
- int [getNumber](#) () const
Returns the course number.
- void [read](#) (std::istream &iStr)
Initializes the objective's members.
- void [write](#) (std::ostream &os) const
Writes the objective to the ostream.

4.28.1 Detailed Description

A subclass of [Objective](#)

4.28.2 Member Function Documentation

4.28.2.1 void SingleObjective::read (std::istream & *iStr*) [virtual]

Should be in the format:

[department] [number]

Implements [Objective](#).

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

- [SingleObjective.h](#)
- [SingleObjective.cc](#)

4.29 SingleObjectiveTest Class Reference

class to test functionality of the [SingleObjective](#) class

```
#include <SingleObjectiveTest.h>
```

Public Member Functions

- void [setUp](#) ()
override setUp to create variable
- void [tearDown](#) ()
override tearDown to clean up

4.29.1 Member Function Documentation

4.29.1.1 void SingleObjectiveTest::setUp ()

Create some variables to test with.

4.29.1.2 void SingleObjectiveTest::tearDown ()

Free space used by allocated variables.

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

- [SingleObjectiveTest.h](#)
- [SingleObjectiveTest.cc](#)

4.30 Student Class Reference

A class to represent a student's course status.

```
#include <Student.h>
```

Public Member Functions

- **Student** ([Major](#) *m=NULL)
- [Major](#) * [getMajor](#) () const
Returns a pointer to the student's major.
- const std::vector< [Course](#) * > [getCourses](#) () const
Returns the vector of course pointers.
- void [write](#) (std::ostream &os) const
Prints the student's members.
- void [changeMajor](#) ([Major](#) *m)
Changes the student's major.
- void [addCourse](#) ([Course](#) *c) throw (CourseAlreadyExists)
Adds a pointer to a course in cl to coursesTaken.
- void [removeCourse](#) (const [Course](#) *c) throw (CourseNotFound)
Removes the pointer to c from coursesToTake.
- bool [status](#) (std::ostream &os) const
Calls major's status function with coursesTaken.
- bool [canTake](#) (const [Course](#) *c) const
Call's c's Objectives' statuses with coursesTaken.

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

- [Student.h](#)
- [Student.cc](#)

4.31 StudentNotFound Class Reference

Exception class to handle invalid student names.

```
#include <Exceptions.h>
```

Inherits `std::runtime_error`.

Public Member Functions

- [StudentNotFound](#) (const std::string &arg)

Constructor.

4.31.1 Constructor & Destructor Documentation

4.31.1.1 StudentNotFound::StudentNotFound (const std::string & *arg*) [inline]

Parameters:

arg the message to be displayed

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

- [Exceptions.h](#)

4.32 StudentTest Class Reference

class to test functionality of the [Student](#) class

```
#include <StudentTest.h>
```

Public Member Functions

- void [setUp](#) ()
override `setUp` to create variable
- void [tearDown](#) ()
override `tearDown` to clean up
- void [initCourseVectors](#) ()
Adds some sample [Course](#) data to some vectors so that we can initialize a [Student](#).
- void [testAddCourse](#) ()
Tests that the [Student::addCourse](#) function throws the correct exceptions, if applicable.
- void [testCanTake](#) ()
Tests that a [Student](#) either can or cannot take a given course.
- void [testChangeMajor](#) ()
Tests that it is possible for a [Student](#) to change his or her major.
- void [testGetCourses](#) ()
Tests that the Courses that the [Student](#) has taken correspond exactly to the Courses that were added.
- void [testGetMajor](#) ()
Tests that the [Student](#) will return its correct [Major](#).
- void [testRemoveCourse](#) ()
Tests that a [Course](#) can be removed from the [Student](#).
- void [testStatus](#) ()
Tests that the status of the [Student](#) is correct after adding some Courses, or changing the [Major](#), and so on.
- void [testWrite](#) ()
Tests that the correct information is written to the output stream when [Student::write](#) is called.

4.32.1 Member Function Documentation

4.32.1.1 void StudentTest::setUp ()

Create some variables to test with.

References initCourseVectors().

4.32.1.2 void StudentTest::tearDown ()

Free space used by allocated variables.

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

- [StudentTest.h](#)
- [StudentTest.cc](#)

Chapter 5

File Documentation

5.1 Course.cc File Reference

A class to represent a University of Lethbridge course. `#include "Course.h"`

Functions

- `std::istream & operator>> (std::istream &iStr, Course &c)`
Overload the extraction operator.
- `std::ostream & operator<< (std::ostream &os, const Course &c)`
Overload the insertion operator.
- `const bool operator== (const Course &firstCourse, const Course &secondCourse)`
Overload the equality test operator.

5.1.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.2 Course.h File Reference

```
A class to represent a University of Lethbridge course. #include <cstdlib>
#include <iostream>
#include <string>
#include <vector>
#include "Exceptions.h"
#include "Course.h"
#include "GLER.h"
#include "Objective.h"
#include "OneOfObjective.h"
#include "SingleObjective.h"
```

Classes

- class [Course](#)

A class to represent a University of Lethbridge course.

Functions

- `std::istream & operator>> (std::istream &iStr, Course &c)`
Overload the extraction operator.
- `std::ostream & operator<< (std::ostream &os, const Course &c)`
Overload the insertion operator.
- `const bool operator== (const Course &firstCourse, const Course &secondCourse)`
Overload the equality test operator.

5.2.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.3 CourseList.cc File Reference

A master list of courses. `#include "CourseList.h"`

Functions

- `std::istream & operator>> (std::istream &iStr, CourseList &cl)`
Overload the extraction operator.
- `std::ostream & operator<< (std::ostream &os, const CourseList &cl)`
Overload the insertion operator.

5.3.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.4 CourseList.h File Reference

```
A master list of courses. #include <fstream>
#include <vector>
#include "Exceptions.h"
#include "Course.h"
```

Classes

- class [CourseList](#)
A master list of courses.

Functions

- `std::istream & operator>> (std::istream &iStr, CourseList &cl)`
Overload the extraction operator.
- `std::ostream & operator<< (std::ostream &os, const CourseList &cl)`
Overload the insertion operator.

5.4.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.5 CourseListTest.cc File Reference

Implementation of the unit tests for the [CourseList](#) class. `#include "CourseListTest.h"`

5.5.1 Detailed Description

Author:

Christopher Rabl

Date:

November 30, 2010

5.6 CourseListTest.h File Reference

Unit tests for the [CourseList](#) class. `#include <cppunit/TestFixture.h>`
`#include <cppunit/extensions/HelperMacros.h>`
`#include <fstream>`
`#include "Exceptions.h"`
`#include "CourseList.h"`

Classes

- class [CourseListTest](#)
class to test functionality of the [CourseList](#) class

5.6.1 Detailed Description

Author:

Christopher Rabl 001051542

Date:

November 30, 2010

5.7 CourseTest.cc File Reference

Tests the member functions of [Course](#) to ensure proper functionality. `#include "CourseTest.h"`

5.7.1 Detailed Description

Author:

Christopher Rabl 001051542

Date:

November 22, 2010

5.8 CourseTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include <fstream>
#include <iostream>
#include <sstream>
#include "Exceptions.h"
#include "Course.h"
```

Classes

- class [CourseTest](#)
class to test functionality of the [Course](#) class

5.8.1 Detailed Description

Author:

Christopher Rabl 001051542

Date:

November 30, 2010

5.9 Exceptions.h File Reference

Exception library for the application. `#include <stdexcept>`
`#include <string>`

Classes

- class [InvalidFile](#)
Exception class to handle invalid file names.
- class [StudentNotFound](#)
Exception class to handle invalid student names.
- class [CourseNotFound](#)
Exception class to handle invalid course names.
- class [InvalidObjective](#)
Exception class to handle invalid Objectives.
- class [CourseAlreadyExists](#)
Exception thrown when a duplicate course is being read into a [CourseList](#).
- class [MajorAlreadyExists](#)
Exception thrown when a duplicate major is being read into a [MajorList](#).
- class [InvalidGLER](#)
Exception thrown when an invalid GLER is encountered.
- class [InvalidPlan](#)
Exception thrown when an invalid [Plan](#) is created or encountered.

5.9.1 Detailed Description

Author:

Christopher Rabl 001051542, Steven Henke 001141476

Date:

November 9, 2010

5.10 FortyCoursesObjective.cc File Reference

The main objective for completing a generic major. `#include
"FortyCoursesObjective.h"`

5.10.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.11 FortyCoursesObjective.h File Reference

The main objective for completing a generic major. `#include <iostream>`

```
#include <vector>
```

```
#include "Exceptions.h"
```

```
#include "Course.h"
```

```
#include "Objective.h"
```

Classes

- class [FortyCoursesObjective](#)

An objective requiring 40 courses to be taken.

5.11.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.12 FortyCoursesObjectiveTest.cc File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "FortyCoursesObjective.h"
#include "FortyCoursesObjectiveTest.h"
```

5.12.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.13 FortyCoursesObjectiveTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "FortyCoursesObjective.h"
```

Classes

- class [FortyCoursesObjectiveTest](#)
class to test functionality of the [FortyCoursesObjective](#) class

5.13.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.14 GLER.h File Reference

An enumerated data type, either SCIENCE, SOCIAL_SCIENCE, or FINE_ARTS_HUMANITY. `#include "Exceptions.h"`

`#include <iostream>`

Enumerations

- enum [GLER](#) { SCIENCE, SOCIAL_SCIENCE, FINE_ARTS_HUMANITY }

Defines the GLER categories; used in all Courses.

Functions

- `std::istream & operator>> (std::istream &iStr, GLER &g) throw (InvalidGLER)`

Overload the extraction operator.

- `std::ostream & operator<< (std::ostream &os, const GLER &g)`

Overload the insertion operator.

5.14.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.15 GLERObjective.cc File Reference

An objective requiring 4 of a particular GLER category to be taken. `#include "GLERObjective.h"`

5.15.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.16 GLERObjective.h File Reference

An objective requiring 4 of a particular GLER category to be taken. `#include <vector>`

```
#include "Exceptions.h"
```

```
#include "Course.h"
```

```
#include "GLER.h"
```

```
#include "Objective.h"
```

Classes

- class [GLERObjective](#)

An objective requiring 4 of a particular GLER category to be taken.

5.16.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.17 GLERObjectiveTest.cc File Reference

```
#include "GLERObjectiveTest.h"
```

5.17.1 Detailed Description

Author:

Christopher Rabl 001051542

Date:

November 22, 2010

5.18 GLERObjectiveTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include <iostream>
#include <sstream>
#include <vector>
#include "Course.h"
#include "GLERObjective.h"
```

Classes

- class [GLERObjectiveTest](#)
class to test functionality of the [GLERObjective](#) class

5.18.1 Detailed Description

Author:

Bradley Ellert 001133286 Christopher Rabl 001051542 Steven Henke 001141476

Date:

November 22, 2010

5.19 GLERTest.cc File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "GLER.h"
#include "GLERTest.h"
```

5.19.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.20 GLERTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "GLER.h"
```

Classes

- class [GLERTest](#)
class to test functionality of the GLER class

5.20.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.21 Major.cc File Reference

A class to represent a University of Lethbridge major. `#include "Major.h"`

Functions

- `std::istream & operator>> (std::istream &iStr, Major &m)`
Overload the extraction operator.
- `std::ostream & operator<< (std::ostream &os, const Major &m)`
Overload the insertion operator.

5.21.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.22 Major.h File Reference

A class to represent a University of Lethbridge major. `#include <iostream>`

`#include <string>`

`#include <vector>`

`#include "Exceptions.h"`

`#include "Objective.h"`

`#include "FortyCoursesObjective.h"`

`#include "GLERObjective.h"`

`#include "SingleObjective.h"`

`#include "MinimumObjective.h"`

`#include "OneOfObjective.h"`

Classes

- class [Major](#)

A class to represent a University of Lethbridge major.

Functions

- `std::istream & operator>> (std::istream &iStr, Major &m)`
Overload the extraction operator.
- `std::ostream & operator<< (std::ostream &os, const Major &m)`
Overload the insertion operator.

5.22.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.23 MajorList.cc File Reference

A master list of majors. `#include "MajorList.h"`

Functions

- `std::istream & operator>> (std::istream &iStr, MajorList &ml)`
Overload of the >> operator for MajorList.
- `std::ostream & operator<< (std::ostream &os, const MajorList &ml)`
Overload of the << operator for MajorList.

5.23.1 Detailed Description

Author:

Bradley Ellert 001133286 Christopher Rabl 001051542 Steven Henke 001141476

Date:

November 22, 2010

5.23.2 Function Documentation

5.23.2.1 `std::ostream& operator<< (std::ostream &os, const MajorList &ml)`

Overload the insertion operator.

References `MajorList::write()`.

5.23.2.2 `std::istream& operator>> (std::istream &iStr, MajorList &ml)`

Overload the extraction operator.

References `MajorList::read()`.

5.24 MajorList.h File Reference

```
A master list of majors. #include <fstream>
#include <string>
#include <vector>
#include "Exceptions.h"
#include "Major.h"
```

Classes

- class [MajorList](#)
A master list of majors.

Functions

- `std::istream & operator>> (std::istream &iStr, MajorList &ml)`
Overload the extraction operator.
- `std::ostream & operator<< (std::ostream &os, const MajorList &ml)`
Overload the insertion operator.

5.24.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.25 MajorListTest.cc File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "MajorList.h"
#include "MajorListTest.h"
```

5.25.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.26 MajorListTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "MajorList.h"
```

Classes

- class [MajorListTest](#)
class to test functionality of the [MajorList](#) class

5.26.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.27 MajorTest.cc File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "Major.h"
#include "MajorTest.h"
```

5.27.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.28 MajorTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "Major.h"
```

Classes

- class [MajorTest](#)
class to test functionality of the [Major](#) class

5.28.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.29 MinimumObjective.cc File Reference

An [Objective](#) requiring a given number of courses at a particular level to be completed.
`#include "MinimumObjective.h"`

5.29.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.30 MinimumObjective.h File Reference

An [Objective](#) requiring a given number of courses at a particular level to be completed.

```
#include <iostream>
```

```
#include <string>
```

```
#include <vector>
```

```
#include "Exceptions.h"
```

```
#include "Course.h"
```

```
#include "Objective.h"
```

Classes

- class [MinimumObjective](#)

An [Objective](#) requiring a given number of courses at a particular level to be completed.

5.30.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.31 MinimumObjectiveTest.cc File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "MinimumObjective.h"
#include "MinimumObjectiveTest.h"
```

5.31.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.32 MinimumObjectiveTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "MinimumObjective.h"
```

Classes

- class [MinimumObjectiveTest](#)
class to test functionality of the [MinimumObjective](#) class

5.32.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.33 Objective.h File Reference

An abstract base class to define the requirements of a major and track their completion status. `#include <iostream>`

`#include <vector>`

Classes

- class [Objective](#)

An abstract base class to define the requirements of a major and track their completion status.

5.33.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.34 OneOfObjective.cc File Reference

An objective requiring a single [Course](#) out of a list of Courses to be completed.
`#include "OneOfObjective.h"`

5.34.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.35 OneOfObjective.h File Reference

An objective requiring a single [Course](#) out of a list of Courses to be completed.

```
#include <iostream>

#include <vector>

#include "Exceptions.h"

#include "Course.h"

#include "Objective.h"
```

Classes

- class [OneOfObjective](#)

An objective requiring a single [Course](#) out of a list of Courses to be completed.

5.35.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.36 OneOfObjectiveTest.cc File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "OneOfObjective.h"
#include "OneOfObjectiveTest.h"
```

5.36.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.37 OneOfObjectiveTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "OneOfObjective.h"
```

Classes

- class [OneOfObjectiveTest](#)
class to test functionality of the [OneOfObjective](#) class

5.37.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.38 Plan.cc File Reference

Builds a potential future plan for a [Student](#). `#include "Plan.h"`

Functions

- `std::ostream & operator<< (std::ostream &os, const Plan &p)`
Overload the insertion operator.

5.38.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.39 Plan.h File Reference

Builds a potential future plan for a [Student](#). `#include <iostream>`

```
#include <sstream>
```

```
#include <vector>
```

```
#include "Exceptions.h"
```

```
#include "Course.h"
```

```
#include "Major.h"
```

```
#include "Student.h"
```

Classes

- class [Plan](#)

Builds a potential future plan for a [Student](#).

Functions

- `std::ostream & operator<< (std::ostream &os, const Plan &p)`

Overload the insertion operator.

5.39.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.40 PlanTest.cc File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "PlanTest.h"
```

5.40.1 Detailed Description

5.41 PlanTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "Plan.h"
```

Classes

- class [PlanTest](#)
class to test functionality of the [Plan](#) class

5.41.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.42 SingleObjective.cc File Reference

An objective requiring one, specific course to be completed. `#include "SingleObjective.h"`

5.42.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.43 SingleObjective.h File Reference

An objective requiring one, specific course to be completed. `#include <iostream>`

`#include <vector>`

`#include "Exceptions.h"`

`#include "Course.h"`

`#include "Objective.h"`

Classes

- class [SingleObjective](#)

An objective requiring one, specific course to be completed.

5.43.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.44 SingleObjectiveTest.cc File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "SingleObjective.h"
#include "SingleObjectiveTest.h"
```

5.44.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.45 SingleObjectiveTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "SingleObjective.h"
```

Classes

- class [SingleObjectiveTest](#)
class to test functionality of the [SingleObjective](#) class

5.45.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.46 Student.cc File Reference

A class to represent a student's course status. `#include "Student.h"`

Functions

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

Overload the insertion operator.

5.46.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.47 Student.h File Reference

A class to represent a student's course status. `#include <iostream>`

`#include <sstream>`

`#include <vector>`

`#include "Exceptions.h"`

`#include "Course.h"`

`#include "Major.h"`

Classes

- class [Student](#)

A class to represent a student's course status.

Functions

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

Overload the insertion operator.

5.47.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 6, 2010

5.48 StudentTest.cc File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include "Student.h"
#include "StudentTest.h"
```

5.48.1 Detailed Description

Author:

Bradley Ellert 001133286 Christopher Rabl 001051542 Steven Henke 001141476

Date:

November 22, 2010

5.49 StudentTest.h File Reference

```
#include <cppunit/TestFixture.h>
#include <cppunit/extensions/HelperMacros.h>
#include <iostream>
#include <sstream>
#include <vector>
#include "Student.h"
```

Classes

- class [StudentTest](#)
class to test functionality of the [Student](#) class

5.49.1 Detailed Description

Author:

Bradley Ellert 001133286

Date:

November 22, 2010

5.50 tester.cc File Reference

```
#include <cppunit/TextTestRunner.h>
#include "CourseTest.h"
#include "CourseListTest.h"
#include "FortyCoursesObjectiveTest.h"
#include "GLERTest.h"
#include "GLERObjectiveTest.h"
#include "MajorTest.h"
#include "MajorListTest.h"
#include "MinimumObjectiveTest.h"
#include "OneOfObjectiveTest.h"
#include "PlanTest.h"
#include "SingleObjectiveTest.h"
#include "StudentTest.h"
```

Functions

- `int main ()`

The simple main function to run the test suites.

5.50.1 Detailed Description

Main program to run test suites

Index

- canTake
 - Plan, [45](#)
- clone
 - Course, [8](#)
 - FortyCoursesObjective, [18](#)
 - GLERObjective, [21](#)
 - MinimumObjective, [36](#)
 - Objective, [39](#)
- copy
 - Major, [29](#)
- Course, [7](#)
 - clone, [8](#)
 - getDepartment, [8](#)
 - getDescription, [8](#)
 - getGler, [8](#)
 - getNumber, [9](#)
 - getPrerequisites, [9](#)
 - getTitle, [9](#)
 - hasTaken, [9](#)
 - read, [9](#)
- Course.cc, [55](#)
- Course.h, [56](#)
- CourseAlreadyExists, [11](#)
 - CourseAlreadyExists, [11](#)
- CourseList, [12](#)
 - destroy, [12](#)
 - find, [13](#)
 - read, [13](#)
- CourseList.cc, [58](#)
- CourseList.h, [59](#)
- CourseListTest, [14](#)
 - setUp, [14](#)
 - tearDown, [14](#)
- CourseListTest.cc, [60](#)
- CourseListTest.h, [61](#)
- CourseNotFound, [15](#)
 - CourseNotFound, [15](#)
- CourseTest, [16](#)
 - setUp, [16](#)
 - tearDown, [16](#)
 - testCourseRead, [16](#)
- CourseTest.cc, [62](#)
- CourseTest.h, [63](#)
- destroy
 - CourseList, [12](#)
 - MajorList, [32](#)
- Exceptions.h, [64](#)
- find
 - CourseList, [13](#)
 - MajorList, [32](#)
- FortyCoursesObjective, [18](#)
 - clone, [18](#)
 - read, [19](#)
 - status, [19](#)
- FortyCoursesObjective.cc, [65](#)
- FortyCoursesObjective.h, [66](#)
- FortyCoursesObjectiveTest, [20](#)
 - setUp, [20](#)
 - tearDown, [20](#)
- FortyCoursesObjectiveTest.cc, [67](#)
- FortyCoursesObjectiveTest.h, [68](#)
- getDepartment
 - Course, [8](#)
- getDescription
 - Course, [8](#)
- getGler
 - Course, [8](#)
- getInstance
 - MajorList, [33](#)
- getNumber
 - Course, [9](#)

- getPrerequisites
 - Course, 9
- getTitle
 - Course, 9
- GLER.h, 69
- GLERObjective, 21
 - clone, 21
 - read, 21
- GLERObjective.cc, 70
- GLERObjective.h, 71
- GLERObjectiveTest, 23
 - setUp, 23
 - tearDown, 23
- GLERObjectiveTest.cc, 72
- GLERObjectiveTest.h, 73
- GLERTest, 24
 - setUp, 24
 - tearDown, 24
- GLERTest.cc, 74
- GLERTest.h, 75
- hasTaken
 - Course, 9
- initCourseVectors
 - PlanTest, 46
- InvalidFile, 25
 - InvalidFile, 25
- InvalidGLER, 26
 - InvalidGLER, 26
- InvalidObjective, 27
 - InvalidObjective, 27
- InvalidPlan, 28
 - InvalidPlan, 28
- Major, 29
 - copy, 29
 - read, 29
- Major.cc, 76
- Major.h, 77
- MajorAlreadyExists, 31
 - MajorAlreadyExists, 31
- MajorList, 32
 - destroy, 32
 - find, 32
 - getInstance, 33
 - read, 33
 - write, 33
- MajorList.cc, 78
 - operator<<, 78
 - operator>>, 78
- MajorList.h, 79
- MajorListTest, 34
 - setUp, 34
 - tearDown, 34
- MajorListTest.cc, 80
- MajorListTest.h, 81
- MajorTest, 35
 - setUp, 35
 - tearDown, 35
- MajorTest.cc, 82
- MajorTest.h, 83
- MinimumObjective, 36
 - clone, 36
 - read, 36
- MinimumObjective.cc, 84
- MinimumObjective.h, 85
- MinimumObjectiveTest, 38
 - setUp, 38
 - tearDown, 38
- MinimumObjectiveTest.cc, 86
- MinimumObjectiveTest.h, 87
- Objective, 39
 - clone, 39
 - read, 40
- Objective.h, 88
- OneOfObjective, 41
 - read, 42
- OneOfObjective.cc, 89
- OneOfObjective.h, 90
- OneOfObjectiveTest, 43
 - setUp, 43
 - tearDown, 43
- OneOfObjectiveTest.cc, 91
- OneOfObjectiveTest.h, 92
- operator<<
 - MajorList.cc, 78
- operator>>
 - MajorList.cc, 78
- Plan, 44

- canTake, 45
 - status, 45
- Plan.cc, 93
- Plan.h, 94
- PlanTest, 46
 - initCourseVectors, 46
 - setUp, 46
 - tearDown, 46
- PlanTest.cc, 95
- PlanTest.h, 96
- read
 - Course, 9
 - CourseList, 13
 - FortyCoursesObjective, 19
 - GLERObjective, 21
 - Major, 29
 - MajorList, 33
 - MinimumObjective, 36
 - Objective, 40
 - OneOfObjective, 42
 - SingleObjective, 49
- setUp
 - CourseListTest, 14
 - CourseTest, 16
 - FortyCoursesObjectiveTest, 20
 - GLERObjectiveTest, 23
 - GLERTest, 24
 - MajorListTest, 34
 - MajorTest, 35
 - MinimumObjectiveTest, 38
 - OneOfObjectiveTest, 43
 - PlanTest, 46
 - SingleObjectiveTest, 50
 - StudentTest, 54
- SingleObjective, 48
 - read, 49
- SingleObjective.cc, 97
- SingleObjective.h, 98
- SingleObjectiveTest, 50
 - setUp, 50
 - tearDown, 50
- SingleObjectiveTest.cc, 99
- SingleObjectiveTest.h, 100
- status
 - FortyCoursesObjective, 19
 - Plan, 45
- Student, 51
- Student.cc, 101
- Student.h, 102
- StudentNotFound, 52
 - StudentNotFound, 52
- StudentTest, 53
 - setUp, 54
 - tearDown, 54
- StudentTest.cc, 103
- StudentTest.h, 104
- tearDown
 - CourseListTest, 14
 - CourseTest, 16
 - FortyCoursesObjectiveTest, 20
 - GLERObjectiveTest, 23
 - GLERTest, 24
 - MajorListTest, 34
 - MajorTest, 35
 - MinimumObjectiveTest, 38
 - OneOfObjectiveTest, 43
 - PlanTest, 46
 - SingleObjectiveTest, 50
 - StudentTest, 54
- testCourseRead
 - CourseTest, 16
- tester.cc, 105
- write
 - MajorList, 33