

Lab 1

1.1

Generated by Doxygen 1.8.9.1

Thu Mar 12 2015 03:33:33

Contents

1	Todo List	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	Benchmark Class Reference	7
4.1.1	Detailed Description	7
4.1.2	Member Function Documentation	7
4.1.2.1	generateRaport	7
4.1.2.2	getAvr	8
4.1.2.3	measureTime	8
4.1.2.4	test	9
4.1.3	Member Data Documentation	9
4.1.3.1	testTimes	9
4.2	InputFiles Class Reference	9
4.2.1	Detailed Description	10
4.2.2	Constructor & Destructor Documentation	10
4.2.2.1	InputFiles	10
4.2.2.2	InputFiles	10
4.2.3	Member Function Documentation	11
4.2.3.1	generate_random_int_data	11
4.2.3.2	return_file_name	11
4.2.3.3	return_file_size	12
4.2.3.4	return_number_files	12
4.2.3.5	show_info	12
4.2.4	Member Data Documentation	12
4.2.4.1	filesNamesTab	12
4.2.4.2	filesNumber	12

4.2.4.3	filesSizes	13
5	File Documentation	15
5.1	src/benchmark_frm.cpp File Reference	15
5.1.1	Detailed Description	15
5.2	src/benchmark_frm.h File Reference	15
5.2.1	Detailed Description	15
5.2.2	Variable Documentation	16
5.2.2.1	SEC	16
5.3	src/inputfile_txt.cpp File Reference	16
5.3.1	Detailed Description	16
5.4	src/inputfile_txt.h File Reference	16
5.4.1	Detailed Description	16
5.4.2	Variable Documentation	16
5.4.2.1	FIRST_ARGUMENT	16
5.4.2.2	PROGRAM_NAME	17
5.4.2.3	UNDEF_VALUE	17
5.5	src/main.cpp File Reference	17
5.5.1	Detailed Description	17
5.5.2	Function Documentation	17
5.5.2.1	main	17
Index		19

Chapter 1

Todo List

Member `Benchmark::generateRaport` (long double nextTime, int size)

FILE OVERWRITING, NEED TO IMPLEMENT NEW NAMES

Member `InputFiles::InputFiles` ()

EXCEPTIONS HANDLING

Chapter 2

Class Index

2.1 Class List

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

Benchmark	7
InputFiles	9

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

src/ benchmark_frm.cpp	
Source code for Benchmark class	15
src/ benchmark_frm.h	
A Benchmark class	15
src/ inputfile_txt.cpp	
Source code for InputFile class	16
src/ inputfile_txt.h	
A InputFile class	16
src/ main.cpp	
Master file	17

Chapter 4

Class Documentation

4.1 Benchmark Class Reference

```
#include <benchmark_frm.h>
```

Public Member Functions

- void [test](#) ([InputFiles](#) files)
Main testing function.

Private Member Functions

- void [generateRaport](#) (long double nextTime, int size)
Generates raport with program tests outputs.
- long double [getAvr](#) (std::vector< long double >times)
Measures the average duration from 10 samples.
- void [measureTime](#) (int *dataTable, int dataSize)
Measures the duration of the work of assignment function.

Private Attributes

- std::vector< long double > [testTimes](#)
A container for calculated times.

4.1.1 Detailed Description

Making a framework for testing inserted data structure. Using time to estimate computational complexity.

4.1.2 Member Function Documentation

4.1.2.1 [Benchmark::generateRaport](#) (long double *nextTime*, int *size*) [private]

Generates raport with program tests outputs.

Parameters

<i>nextTime</i>	A new calculated time (for new file size).
<i>Size</i>	A size of the currently working file.

Todo FILE OVERWRITING, NEED TO IMPLEMENT NEW NAMES

```

6                                     {
7     /*! \todo FILE OVERWRITING, NEED TO IMPLEMENT NEW NAMES */
8     std::ofstream raportFile;
9     std::string stringNextTime = std::to_string(nextTime);
10
11     // .xls as excel file format
12     raportFile.open("test.xls", std::ios::in | std::ios::app);
13     assert(raportFile.is_open() && ("I can't open file.));
14     // need to change '.' on ',' due to excel string format
15     boost::algorithm::replace_first(stringNextTime, ".", ",");
16
17     raportFile << size << "\n" << stringNextTime << "\n";
18     raportFile.close();
19 }
```

4.1.2.2 `Benchmark::getAvr (std::vector< long double > times)` [private]

Measures the average duration from 10 samples.

Parameters

<i>times</i>	A container with times from tests.
--------------	------------------------------------

```

21                                     {
22     long double avrg = 0.0;
23
24     //add 10 values, than count average
25     for (int i = 0; i < (signed)times.size(); i++){
26         avrg += times[i];
27     }
28
29     avrg /= (long double)times.size();
30     return avrg;
31 }
```

4.1.2.3 `Benchmark::measureTime (int * dataTable, int dataSize)` [private]

Measures the duration of the work of assignment function.

Parameters

<i>dataTable</i>	A container with random integers from earlier made files.
<i>dataSize</i>	A size of the file.

```

33                                     {
34     // container for counted working times
35     std::vector<long double> estimateTimes;
36
37     for (int j = 0; j < 10; j++){
38         // Here starts the timer
39         boost::timer::cpu_timer startTime;
40         for (int i = 0; i < dataSize; i++){
41             dataTable[i] *= 2;
42         }
43         // Here it ends
44         boost::timer::cpu_times endTime = startTime.elapsed();
45         // add new time to the vector
46         estimateTimes.push_back(static_cast<long double>(endTime.wall * SEC));
47     }
48     // for better display
49     std::cout.fixed;
50     long double DurTime = getAvr(estimateTimes);
51     std::cout << "Time (average, 10 samples) for " << dataSize << " elements: " << DurTime << " sec"<<
52     std::endl;
53     generateRapor(DurTime, dataSize);
54 }
```

4.1.2.4 Benchmark::test (InputFiles files)

Main testing function.

Parameters

<i>files</i>	random generated files with integers
--------------	--------------------------------------

```

55                                     {
56     // temp memory container
57     int* tabForData = NULL;
58     int tempValue = 0;
59     int count = 0;
60     std::fstream newFile;
61
62     for (int i = 0; i < files.return_number_files() -
FIRST_ARGUMENT; i++){
63         // Opening file + making new table with content
64         tabForData = new int[files.return_file_size(i)];
65         newFile.open((files.return_file_name(i) + ".txt"), std::ios::in);
66
67         // Checking if file is opened correctly
68         assert(newFile.is_open() && ("I can't open file.));
69
70         for (int j = 0; j < files.return_file_size(i); j++){
71             newFile >> tempValue;
72             tabForData[j] = tempValue;
73         }
74         newFile.close();
75
76         // Testing time here
77         measureTime(tabForData, files.return_file_size(i));
78         delete[] tabForData;
79     }
80 }
```

4.1.3 Member Data Documentation

4.1.3.1 Benchmark::testTimes [private]

A container for calculated times.

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

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

4.2 InputFiles Class Reference

```
#include <inputfile_txt.h>
```

Public Member Functions

- void [generate_random_int_data](#) ()
Puts random int data into files.
- [InputFiles](#) ()
A default constructor.
- [InputFiles](#) (int filNr, std::vector< int >filSiz)
A constructor.
- const std::string [return_file_name](#) (int Nmbr)
Return names of files (only for read purpose)
- const int [return_file_size](#) (int Nmbr)
Return sizes of files (only for read purpose)

- const int `return_number_files` ()
Return number of files.
- void `show_info` ()
Showes info about files.

Private Attributes

- std::vector< std::string > `filesNamesTab`
Container for generated file names.
- int `filesNumber`
Number of generated files.
- std::vector< int > `filesSizes`
Container for file sizes.

4.2.1 Detailed Description

Making an object which contain text files with generated random integer numbers.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 InputFiles::InputFiles ()

A default constructor.

Adding number of files(UNDEF_VALUE = 1); Generating file name; Adding size of file (UNDEF_VALUE = 1);

Just in case, when program starts without any parameters.

Todo EXCEPTIONS HANDLING

```

6         {
7     /*! \todo EXCEPTIONS HANDLING */
8     // When there are no arguments from command prompt:
9     filesNumber = UNDEF_VALUE;
10    std::string TempName = std::tmpnam(nullptr);
11    filesNamesTab.push_back(TempName);
12    filesSizes.push_back(UNDEF_VALUE);
13 }
```

4.2.2.2 InputFiles::InputFiles (int filNr, std::vector< int > filSiz)

A constructor.

Adding number of files; Generating files names; Adding sizes of files; Parameters inherit from list of arguments from command prompt

Parameters

<i>filNr</i>	number of files
<i>filSiz</i>	sizes of files

Open files with new names

Check if file is opened correctly

```

15         {
16     filesNumber = filNr;
17     filesSizes = filSiz;
18     // Create new names for files
19     std::string TempName;
```

```

20
21     for (int i = 1; i < filesNumber; i++){
22         // Generate new unique name for file
23         TempName = std::tmpnam(nullptr);
24         // Delete all prohibit char from string
25         boost::algorithm::erase_all(TempName, "/");
26         boost::algorithm::erase_all(TempName, "\\");
27         // Put name into names container
28         filesNamesTab.push_back(TempName);
29     }
30     //! Open files with new names
31     std::ofstream newFile;
32     for (int i = 1; i < filesNumber; i++){
33         newFile.open(filesNamesTab[i - PROGRAM_NAME] + ".txt");
34         //! Check if file is opened correctly
35         assert(newFile.is_open() && "I can't open this file.");
36         newFile.close();
37     }
38 }

```

4.2.3 Member Function Documentation

4.2.3.1 InputFiles::generate_random_int_data ()

Puts random int data into files.

Generating random integers data (size from filesSizes vector) and putting them into files (names from filesNamesTab) < Seed for Mersenne Twister 19937 generator

Mersenne Twister 19937 generator

More info about this generator: http://pl.wikipedia.org/wiki/Mersenne_Twister

Uniform distribution random number

Max number: uncomment next line More info about this distribution: http://pl.wikipedia.org/wiki/Rozk%C5%82ad_jednostajny

Check if file is opened correctly

```

52     {
53         int seedGen = time(NULL); /*!< Seed for Mersenne Twister 19937 generator */
54
55         //! Mersenne Twister 19937 generator
56         /*!
57             More info about this generator:
58             <a href="linkURL">http://pl.wikipedia.org/wiki/Mersenne_Twister</a>
59         */
60         std::mt19937 randomNumbr(seedGen);
61
62         //! Uniform distribution random number
63         /*!
64             Max number: uncomment next line
65             More info about this distribution:
66             <a href="linkURL">http://pl.wikipedia.org/wiki/Rozk%C5%82ad_jednostajny</a>
67         */
68         //std::cout << std::numeric_limits<int>::max() << std::endl;
69         std::uniform_int_distribution<>newDistr;
70
71         std::ofstream NewFile;
72         for (int i = 1; i < filesNumber; i++){
73             NewFile.open((filesNamesTab[i - PROGRAM_NAME] + ".txt"), std::ios::in);
74             //! Check if file is opened correctly
75             assert(NewFile.is_open() && ("I can't open file."));
76             //Generate random int data
77             for (int j = 0; j < filesSizes[i - FIRST_ARGUMENT]; j++){
78                 NewFile << newDistr(randomNumbr) << "\n";
79             }
80
81             NewFile.close();
82         }
83     }

```

4.2.3.2 InputFiles::return_file_name (int Nmbr) [inline]

Return names of files (only for read purpose)

Parameters

<i>Nmbr</i>	Number of the file.
-------------	---------------------

```

70                                     {
71         return filesNamesTab[Nmbr];
72     }

```

4.2.3.3 InputFiles::return_file_size (int *Nmbr*) [inline]

Return sizes of files (only for read purpose)

Parameters

<i>Nmbr</i>	Number of the file.
-------------	---------------------

```

78                                     {
79         return filesSizes[Nmbr];
80     }

```

4.2.3.4 InputFiles::return_number_files () [inline]

Return number of files.

```

85                                     {
86         return filesNumber;
87     }

```

4.2.3.5 InputFiles::show_info ()

Showes info about files.

Display: number of files, names of files, sizes of files

```

40         {
41     std::cout << "-----" << std::endl;
42     std::cout << filesNumber - FIRST_ARGUMENT << std::endl;
43     for (int i = 0; i < (signed)filesNamesTab.size(); i++){
44         std::cout << filesNamesTab[i] << std::endl;
45     }
46     for (int i = 0; i < (signed)filesSizes.size(); i++){
47         std::cout << filesSizes[i] << std::endl;
48     }
49     std::cout << "-----" << std::endl;
50 }

```

4.2.4 Member Data Documentation**4.2.4.1 std::vector<std::string> InputFiles::filesNamesTab [private]**

Container for generated file names.

4.2.4.2 InputFiles::filesNumber [private]

Number of generated files.

4.2.4.3 InputFiles::fileSizes [private]

Container for file sizes.

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

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

Chapter 5

File Documentation

5.1 src/benchmark_frm.cpp File Reference

Source code for [Benchmark](#) class.

```
#include "benchmark_frm.h"
```

5.1.1 Detailed Description

Source code for [Benchmark](#) class.

5.2 src/benchmark_frm.h File Reference

A [Benchmark](#) class.

```
#include <vector>
#include <fstream>
#include <boost\timer\timer.hpp>
#include <boost\chrono\duration.hpp>
#include <boost\algorithm\string\replace.hpp>
#include "inputfile_txt.h"
```

Classes

- class [Benchmark](#)

Variables

- const long double [SEC](#) = 0.000000001

5.2.1 Detailed Description

A [Benchmark](#) class.

5.2.2 Variable Documentation

5.2.2.1 `const long double SEC = 0.000000001`

5.3 `src/inputfile_txt.cpp` File Reference

Source code for InputFile class.

```
#include "inputfile_txt.h"
```

5.3.1 Detailed Description

Source code for InputFile class.

5.4 `src/inputfile_txt.h` File Reference

A InputFile class.

```
#include <iostream>
#include <string>
#include <fstream>
#include <vector>
#include <cstdio>
#include <cassert>
#include <ctime>
#include <random>
#include <boost/algorithm/string/erase.hpp>
```

Classes

- class [InputFiles](#)

Variables

- `const int FIRST_ARGUMENT = 1`
*A const value for representing first argument from command prompt (name of the program) */.*
- `const int PROGRAM_NAME = 1`
*The same as FIRST_ARGUMENT */.*
- `const int UNDEF_VALUE = 1`
*A value for undefined arguments */.*

5.4.1 Detailed Description

A InputFile class.

5.4.2 Variable Documentation

5.4.2.1 `const int FIRST_ARGUMENT = 1`

A const value for representing first argument from command prompt (name of the program) */.

5.4.2.2 const int PROGRAM_NAME = 1

The same as FIRST_ARGUMENT */.

5.4.2.3 const int UNDEF_VALUE = 1

A value for undefined arguments */.

5.5 src/main.cpp File Reference

Master file.

```
#include "inputfile_txt.h"
#include "benchmark_frm.h"
```

Functions

- int [main](#) (int argc, char *argv[])

5.5.1 Detailed Description

Master file.

5.5.2 Function Documentation

5.5.2.1 int main (int argc, char * argv[])

```
9 {
10     // Container for sizes from command prompt
11     std::vector<int>FilesSizes;
12
13     // First argument is a name of the program so i = 1
14     for (int i = 1; i < argc ; i++)
15         FilesSizes.push_back(atoi(argv[i]));
16
17     InputFiles newFilesList(argc, FilesSizes);
18     Benchmark NewTest;
19
20     newFilesList.generate_random_int_data();
21     NewTest.test(newFilesList);
22 }
```


Index

- Benchmark, [7](#)
 - generateRaport, [7](#)
 - getAvr, [8](#)
 - measureTime, [8](#)
 - test, [8](#)
 - testTimes, [9](#)
- benchmark_frm.h
 - SEC, [16](#)
- FIRST_ARGUMENT
 - inputfile_txt.h, [16](#)
- filesNamesTab
 - InputFiles, [12](#)
- filesNumber
 - InputFiles, [12](#)
- filesSizes
 - InputFiles, [12](#)
- generate_random_int_data
 - InputFiles, [11](#)
- generateRaport
 - Benchmark, [7](#)
- getAvr
 - Benchmark, [8](#)
- InputFiles, [9](#)
 - filesNamesTab, [12](#)
 - filesNumber, [12](#)
 - filesSizes, [12](#)
 - generate_random_int_data, [11](#)
 - InputFiles, [10](#)
 - return_file_name, [11](#)
 - return_file_size, [12](#)
 - return_number_files, [12](#)
 - show_info, [12](#)
- inputfile_txt.h
 - FIRST_ARGUMENT, [16](#)
 - PROGRAM_NAME, [16](#)
 - UNDEF_VALUE, [17](#)
- main
 - main.cpp, [17](#)
- main.cpp
 - main, [17](#)
- measureTime
 - Benchmark, [8](#)
- PROGRAM_NAME
 - inputfile_txt.h, [16](#)
- return_file_name
 - InputFiles, [11](#)
- return_file_size
 - InputFiles, [12](#)
- return_number_files
 - InputFiles, [12](#)
- SEC
 - benchmark_frm.h, [16](#)
- show_info
 - InputFiles, [12](#)
- src/benchmark_frm.cpp, [15](#)
- src/benchmark_frm.h, [15](#)
- src/inputfile_txt.cpp, [16](#)
- src/inputfile_txt.h, [16](#)
- src/main.cpp, [17](#)
- test
 - Benchmark, [8](#)
- testTimes
 - Benchmark, [9](#)
- UNDEF_VALUE
 - inputfile_txt.h, [17](#)