Lab1

1.0

Generated by Doxygen 1.8.9.1

Wed Mar 11 2015 15:21:46

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 Benchmark Class Reference

`#include <benchmark_frm.h>`

Public Member Functions

- void test (InputFiles files)

3.1.1 Detailed Description

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

3.1.2 Member Function Documentation

3.1.2.1 void Benchmark::test (**InputFiles** ***files***)

Opening file + making new table with content

Check if file is opened correctly

Testing time here

```
                                    {
    int* tabForData = NULL;
    int tempValue = 0;
    int count = 0;
    std::fstream newFile;

    for (int i = 0; i < files.return_number_files() -
   FIRST_ARGUMENT; i++){

        tabForData = new int[files.return_file_size(i)];
        newFile.open((files.return_file_name(i) + ".txt"), std::ios::in);

        assert(newFile.is_open() && ("I can't open file."));

        for (int j = 0; j < files.return_file_size(i); j++){
            newFile >> tempValue;
            tabForData[j] = tempValue;
        }
        newFile.close();

        measureTime(tabForData, files.return_file_size(i));
        delete[] tabForData;

    }
}
```

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

- src/benchmark_frm.h
- src/benchmark_frm.cpp

3.2 InputFiles Class Reference

`#include <inputfile_txt.h>`

Public Member Functions

- InputFiles ()

  A default constructor.
- InputFiles (int filNr, std::vector< int >filSiz)

  A constructor.
- void show_info ()

  Show info about files.
- void generate_random_int_data ()

  Create random integers data into files.
- 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.

3.2.1 Detailed Description

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

3.2.2 Constructor & Destructor Documentation

3.2.2.1 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.

A new file object class source code

```
                     {
   filesNumber = UNDEF_VALUE;

   //TODO: EXCEPTIONS HANDLING
   std::string TempName = std::tmpnam(nullptr);

    filesNamesTab.push_back(TempName);
    filesSizes.push_back(UNDEF_VALUE);
}
```

3.2.2.2 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

filNr	number of files
filSiz	sizes of files

Create new names for files

Delete all prohibit char from string

Open files with new names

Check if file is opened correctly

```
                                                     {
    filesNumber = filNr;
    filesSizes = filSiz;

    std::string TempName;
    for (int i = 1; i < filesNumber; i++){
        TempName = std::tmpnam(nullptr);

        boost::algorithm::erase_all(TempName, "/");
        boost::algorithm::erase_all(TempName, "\\");

        filesNamesTab.push_back(TempName);
    }

    std::ofstream NewFile;
    for (int i = 1; i < filesNumber; i++){
        NewFile.open(filesNamesTab[i - PROGRAM_NAME] + ".txt");
        assert(NewFile.is_open() && "I can't open this file.");
        NewFile.close();
    }
}
```

3.2.3 Member Function Documentation

3.2.3.1 void InputFiles::generate_random_int_data ( )

Create random integers data into files.

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

```
                                         {

    int seedGen = time(NULL);


    std::mt19937 randomNumbr(seedGen);


    //std::cout << std::numeric_limits<int>::max() << std::endl;
    std::uniform_int_distribution<>newDistr;

    std::ofstream NewFile;

    for (int i = 1; i < filesNumber; i++){
        NewFile.open((filesNamesTab[i - PROGRAM_NAME] + ".txt"),std::ios::in);
        assert(NewFile.is_open() && ("I can't open file."));
        for (int j = 0; j < filesSizes[i - FIRST_ARGUMENT]; j++){
            NewFile << newDistr(randomNumbr) << "\n";
        }
        NewFile.close();
    }
}
```

3.2.3.2 const std::string InputFiles::return_file_name (int *Nmbr*) [inline]

Return names of files (only for read purpose)

Parameters

Nmbr	number of the file

```
                                            {
            return filesNamesTab[Nmbr];
        }
```

3.2.3.3 const int InputFiles::return_file_size (int *Nmbr*) [inline]

Return sizes of files (only for read purpose)

Parameters

Nmbr	number of the file

```
                                      {
            return filesSizes[Nmbr];
        }
```

3.2.3.4 const int InputFiles::return_number_files ( ) [inline]

Return number of files.

```
                                 {
            return filesNumber;
        }
```

3.2.3.5 void InputFiles::show_info ( )

Show info about files.

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

```
                          {

    std::cout << "--------" << std::endl;
    std::cout << filesNumber - FIRST_ARGUMENT << std::endl;

    for (int i = 0; i < (signed)filesNamesTab.size(); i++){
        std::cout << filesNamesTab[i] << std::endl;
    }
    for (int i = 0; i < (signed)filesSizes.size(); i++){
        std::cout << filesSizes[i] << std::endl;
    }

    std::cout << "--------" << std::endl;
}
```

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

- src/inputfile_txt.h
- src/inputfile_txt.cpp

Chapter 4

File Documentation

4.1 src/benchmark_frm.cpp File Reference

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

4.2 src/benchmark_frm.h File Reference

```
#include <vector>
#include <fstream>
#include <boost\timer\timer.hpp>
#include "inputfile_txt.h"
```

Classes

- class Benchmark

4.3 src/inputfile_txt.cpp File Reference

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

4.4 src/inputfile_txt.h File Reference

A new input files 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

  First argument from command prompt (name of the program)
- const int UNDEF_VALUE = 1
- const int PROGRAM_NAME = 1

4.4.1 Detailed Description

A new input files class.

4.4.2 Variable Documentation

4.4.2.1 const int FIRST_ARGUMENT = 1

First argument from command prompt (name of the program)

4.4.2.2 const int PROGRAM_NAME = 1

4.4.2.3 const int UNDEF_VALUE = 1

4.5 src/main.cpp File Reference

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

Functions

- int main (int argc, char ∗argv[ ])

4.5.1 Function Documentation

4.5.1.1 int main (int *argc,* char ∗ *argv[ ]*)

Container for sizes from command prompt

First argument is a name of the program so i = 1

```
{
    std::vector<int>FilesSizes;

    for (int i = 1; i < argc ; i++)
        FilesSizes.push_back(atoi(argv[i]));

    InputFiles newFilesList(argc, FilesSizes);
    Benchmark NewTest;
    newFilesList.generate_random_int_data();
    NewTest.test(newFilesList);
}
```

Index