

Project 4 - Laboratory 13 - Performance Evalution - Timer

Generated by Doxygen 1.7.6.1

Wed Sep 24 2014 00:24:09

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	binarySearch Class Reference	7
4.1.1	Member Function Documentation	7
4.1.1.1	operator()	7
4.2	linearSearch Class Reference	7
4.2.1	Member Function Documentation	8
4.2.1.1	operator()	8
4.3	Search Class Reference	8
4.4	STLSearch Class Reference	8
4.4.1	Member Function Documentation	9
4.4.1.1	operator()	9
4.5	TestVector Class Reference	9
4.5.1	Constructor & Destructor Documentation	9
4.5.1.1	TestVector	9
4.5.1.2	TestVector	9
4.5.2	Member Function Documentation	9
4.5.2.1	operator++	9

4.5.2.2	operator++	10
4.5.2.3	operator[]	10
4.6	Timer Class Reference	10
4.6.1	Constructor & Destructor Documentation	10
4.6.1.1	Timer	10
4.6.2	Member Function Documentation	10
4.6.2.1	getElapsedTime	10
4.6.2.2	start	11
4.6.2.3	stop	11
5	File Documentation	13
5.1	config.h File Reference	13
5.1.1	Define Documentation	13
5.1.1.1	LAB13_TEST1	13
5.1.1.2	LAB13_TEST2	13
5.2	constructor.cpp File Reference	13
5.2.1	Define Documentation	14
5.2.1.1	runTest	14
5.2.2	Function Documentation	14
5.2.2.1	main	14
5.2.2.2	testCompute	14
5.2.2.3	testCompute< double >	14
5.2.2.4	testCompute< int >	14
5.2.2.5	testConstructor	14
5.2.3	Variable Documentation	14
5.2.3.1	numRepetitions	14
5.3	inc.cpp File Reference	14
5.3.1	Function Documentation	15
5.3.1.1	main	15
5.3.2	Variable Documentation	15
5.3.2.1	numRepetitions	15
5.4	search.cpp File Reference	15
5.4.1	Function Documentation	15
5.4.1.1	main	15

5.4.2	Variable Documentation	15
5.4.2.1	numSearches	15
5.5	sort.cpp File Reference	15
5.5.1	Function Documentation	16
5.5.1.1	main	16
5.5.1.2	quickSort	16
5.5.1.3	selectionSort	16
5.5.1.4	timeSort	16
5.5.2	Variable Documentation	16
5.5.2.1	numSorts	16
5.6	test13.cpp File Reference	16
5.6.1	Function Documentation	16
5.6.1.1	main	16
5.6.1.2	print_help	16
5.6.1.3	wait	17
5.7	testtimer.cpp File Reference	17
5.7.1	Function Documentation	17
5.7.1.1	getElapsed	17
5.7.1.2	main	17
5.8	testvector.cpp File Reference	17
5.9	testvector.h File Reference	17
5.10	Timer.cpp File Reference	17
5.10.1	Detailed Description	17
5.11	Timer.h File Reference	18

Chapter 1

Class Index

1.1 Class Hierarchy

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

Search	8
binarySearch	7
linearSearch	7
STLSearch	8
TestVector	9
Timer	10

Chapter 2

Class Index

2.1 Class List

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

binarySearch	7
linearSearch	7
Search	8
STLSearch	8
TestVector	9
Timer	10

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

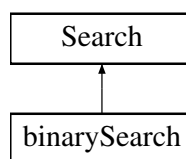
config.h	13
constructor.cpp	13
inc.cpp	14
search.cpp	15
sort.cpp	15
test13.cpp	16
testtimer.cpp	17
testvector.cpp	17
testvector.h	17
Timer.cpp	17
Timer.h	18

Chapter 4

Class Documentation

4.1 `binarySearch` Class Reference

Inheritance diagram for `binarySearch`:



Public Member Functions

- `bool operator\(\) (int searchValue, const vector< int > &keys) const`

4.1.1 Member Function Documentation

4.1.1.1 `bool binarySearch::operator\(\) (int searchValue, const vector< int > & keys) const`
[inline, virtual]

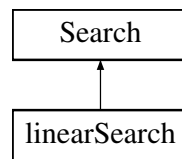
Implements [Search](#).

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

- [search.cpp](#)

4.2 `linearSearch` Class Reference

Inheritance diagram for `linearSearch`:



Public Member Functions

- `bool operator() (int searchValue, const vector< int > &keys) const`

4.2.1 Member Function Documentation

4.2.1.1 `bool linearSearch::operator() (int searchValue, const vector< int > & keys) const`
[inline, virtual]

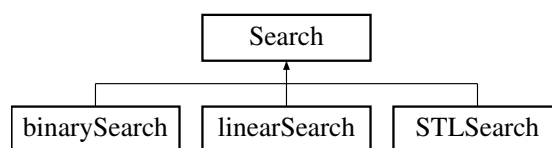
Implements [Search](#).

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

- [search.cpp](#)

4.3 Search Class Reference

Inheritance diagram for Search:

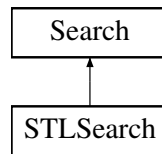


The documentation for this class was generated from the following file:

- [search.cpp](#)

4.4 STLSearch Class Reference

Inheritance diagram for STLSearch:



Public Member Functions

- `bool operator()` (int searchValue, const vector< int > &keys) const

4.4.1 Member Function Documentation

4.4.1.1 `bool STLSearch::operator()` (int *searchValue*, const vector< int > & *keys*) const
[inline, virtual]

Implements [Search](#).

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

- [search.cpp](#)

4.5 TestVector Class Reference

```
#include <testvector.h>
```

Public Member Functions

- `TestVector` (int size)
- `TestVector` (const `TestVector` &rhs)
- `TestVector` & `operator++` ()
- `TestVector operator++` (int ignored)
- int `operator[]` (int loc) const

4.5.1 Constructor & Destructor Documentation

4.5.1.1 `TestVector::TestVector` (int *size*)

4.5.1.2 `TestVector::TestVector` (const `TestVector` & *rhs*)

4.5.2 Member Function Documentation

4.5.2.1 `TestVector` & `TestVector::operator++` ()

4.5.2.2 **TestVector** TestVector::operator++ (int *ignored*)

4.5.2.3 int TestVector::operator[] (int *loc*) const

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

- [testvector.h](#)
- [testvector.cpp](#)

4.6 Timer Class Reference

```
#include <Timer.h>
```

Public Member Functions

- [Timer](#) ()
- void [start](#) () throw (runtime_error)
- void [stop](#) () throw (logic_error)
- double [getElapsedTime](#) () const throw (logic_error)

4.6.1 Constructor & Destructor Documentation

4.6.1.1 **Timer::Timer** ()

Default constructor.

Default constructor of timer class. Sets data members to zero and false.

4.6.2 Member Function Documentation

4.6.2.1 double **Timer::getElapsedTime** () const throw (logic_error)

Returns elapsed time

Calculates elapsed time by converting microseconds to seconds. Returns this value. - Throws a logic error if the duration was never set because the timer had not been turned on.

Exceptions

<i>Throws</i>	logic error if duration not set.
---------------	----------------------------------

throws a logic error if duration was never set

return the value of duration divided by 1000000 to give seconds instead of microseconds

4.6.2.2 void Timer::start () throw (runtime_error)

Start timer.

Begins keeping track of the time passing, until stop. Throws an exception if timer could not start.

Exceptions

<i>throws</i>	error if timer did not start
---------------	------------------------------

gets the time of the day

throw runtime error if timer did not start

set that the timer has been started

4.6.2.3 void Timer::stop () throw (logic_error)

Stops timer.

Stops timer incrementation and calculates duration that timer counter was on.

Exceptions

<i>throws</i>	logic error if timer never started
---------------	------------------------------------

if the timer hasn't been started, throw exception

instantiate time value holder variables

set the current time to the duration

gets the values of the microseconds

sets duration to difference of start and end times

sets that timer is not running anymore

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

- [Timer.h](#)
- [Timer.cpp](#)

Chapter 5

File Documentation

5.1 config.h File Reference

Defines

- `#define LAB13_TEST1 0`
- `#define LAB13_TEST2 0`

5.1.1 Define Documentation

5.1.1.1 `#define LAB13_TEST1 0`

`Timer` class (Lab 13) configuration file. Activate test 'N' by defining the corresponding `LAB12_TESTN` to have the value 1.

5.1.1.2 `#define LAB13_TEST2 0`

5.2 constructor.cpp File Reference

```
#include <iostream> #include <string> #include "Timer.h" ×  
#include "testvector.h"
```

Defines

- `#define runTest(Type) testConstructor<Type>(numValues, #Type)`

Functions

- `template<typename DataType >
int testCompute (DataType value)`

- `template<>`
 `int testCompute< int > (int value)`
- `template<>`
 `int testCompute< double > (double value)`
- `template<typename DataType >`
 `void testConstructor (int numValues, string name)`
- `int main (int argc, char **argv)`

Variables

- `const int numRepetitions = 1000000`

5.2.1 Define Documentation

5.2.1.1 `#define runTest(Type) testConstructor<Type>(numValues, #Type)`

5.2.2 Function Documentation

5.2.2.1 `int main (int argc, char ** argv)`

5.2.2.2 `template<typename DataType > int testCompute (DataType value)`

5.2.2.3 `template<> int testCompute< double > (double value)`

5.2.2.4 `template<> int testCompute< int > (int value)`

5.2.2.5 `template<typename DataType > void testConstructor (int numValues, string name)`

5.2.3 Variable Documentation

5.2.3.1 `const int numRepetitions = 1000000`

5.3 inc.cpp File Reference

```
#include <iostream> #include "Timer.h" #include "testvector.h"
```

Functions

- `int main (int argc, char **argv)`

Variables

- `const int numRepetitions = 1000000`

5.3.1 Function Documentation

5.3.1.1 `int main (int argc, char ** argv)`

5.3.2 Variable Documentation

5.3.2.1 `const int numRepetitions = 1000000`

5.4 search.cpp File Reference

```
#include <iostream> #include <algorithm> #include <vector> ×  
#include "Timer.h"
```

Classes

- class [Search](#)
- class [linearSearch](#)
- class [binarySearch](#)
- class [STLSearch](#)

Functions

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

Variables

- const int [numSearches](#) = 100000

5.4.1 Function Documentation

5.4.1.1 `int main (int argc, char ** argv)`

5.4.2 Variable Documentation

5.4.2.1 `const int numSearches = 100000`

5.5 sort.cpp File Reference

```
#include <iostream> #include <algorithm> #include <vector> ×  
#include "Timer.h"
```

Functions

- void [selectionSort](#) (vector< int >::iterator front, vector< int >::iterator back)
- void [quickSort](#) (vector< int >::iterator front, vector< int >::iterator back)
- void [timeSort](#) (void(*fcn)(vector< int >::iterator front, vector< int >::iterator back), const string name, const vector< int > &masterList, const [Timer](#) &overhead)
- int [main](#) (int argc, char **argv)

Variables

- const int [numSorts](#) = 100

5.5.1 Function Documentation

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

5.5.1.2 void [quickSort](#) (vector< int >::iterator *front*, vector< int >::iterator *back*)

5.5.1.3 void [selectionSort](#) (vector< int >::iterator *front*, vector< int >::iterator *back*)

5.5.1.4 void [timeSort](#) (void(*) (vector< int >::iterator front, vector< int >::iterator back) *fcn*, const string *name*, const vector< int > & *masterList*, const [Timer](#) & *overhead*)

5.5.2 Variable Documentation

5.5.2.1 const int [numSorts](#) = 100

5.6 test13.cpp File Reference

```
#include <iostream> #include <cctype> #include <ctime> ×
#include "Timer.h"
```

Functions

- void [wait](#) (int secs)
- void [print_help](#) ()
- int [main](#) ()

5.6.1 Function Documentation

5.6.1.1 int [main](#) ()

5.6.1.2 void [print_help](#) ()

5.6.1.3 void wait (int secs)

5.7 testtimer.cpp File Reference

```
#include "Timer.h" #include <iostream> #include <stddef.-  
h> #include <sys/time.h> #include <cstdio>
```

Functions

- double [getElapsed](#) (timeval &t1)
- int [main](#) (int argc, char **argv)

5.7.1 Function Documentation

5.7.1.1 double [getElapsed](#) (timeval & t1)

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

5.8 testvector.cpp File Reference

```
#include <functional> #include <algorithm> #include "testvector.-  
h"
```

5.9 testvector.h File Reference

```
#include <stdexcept> #include <iostream> #include <vector> ×
```

Classes

- class [TestVector](#)

5.10 Timer.cpp File Reference

```
#include "Timer.h"
```

5.10.1 Detailed Description

Author

CatherinePollock

Date

9/16/14

This is the file that implements [Timer.h](#) and the [Timer](#) class within that file.

5.11 Timer.h File Reference

```
#include <ctime> #include <sys/time.h> #include <stdexcept> ×  
#include <iostream>
```

Classes

- class [Timer](#)