PA04

Generated by Doxygen 1.8.11

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

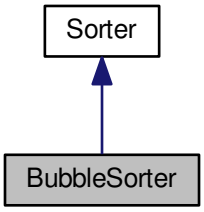
Class Documentation

4.1 BubbleSorter Class Reference

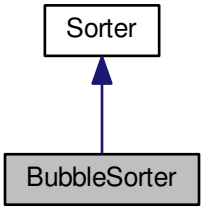
BubbleSorter class.

`#include <BubbleSorter.h>`

Inheritance diagram for BubbleSorter:



Collaboration diagram for BubbleSorter:



Public Member Functions

- BubbleSorter ()
- void sortInPlace (int *arr, int len)

Additional Inherited Members

4.1.1 Detailed Description

BubbleSorter class.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 BubbleSorter::BubbleSorter ( )

Default constructor for Sorters.

Constructs with name: "Bubble Sorter"

4.1.3 Member Function Documentation

4.1.3.1 void BubbleSorter::sortInPlace (int * *arr*, int *len*) [virtual]

Sorts an array in place.

Uses the bubble sort algorithm.

Implements Sorter.

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

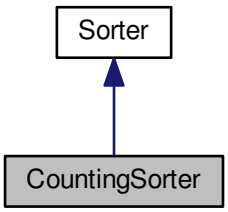
- BubbleSorter.h
- BubbleSorter.cpp

4.2 CountingSorter Class Reference

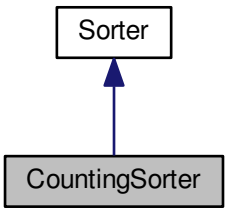
CountingSorter class.

`#include <CountingSorter.h>`

Inheritance diagram for CountingSorter:



Collaboration diagram for CountingSorter:



Public Member Functions

- CountingSorter ()
- void sortInPlace (int *arr, int len)

Additional Inherited Members

4.2.1 Detailed Description

CountingSorter class.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 CountingSorter::CountingSorter ( )

4.2.3 Member Function Documentation

4.2.3.1 void CountingSorter::sortInPlace (int * *arr*, int *len*) [virtual]

Sorts an array in place.

Uses the counting sort algorithm.

Implements Sorter.

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

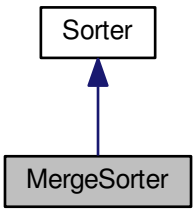
- CountingSorter.h
- CountingSorter.cpp

4.3 MergeSorter Class Reference

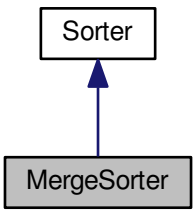
MergeSorter class.

`#include <MergeSorter.h>`

Inheritance diagram for MergeSorter:



Collaboration diagram for MergeSorter:



Public Member Functions

- MergeSorter ()
- void sortInPlace (int *arr, int len)

Private Member Functions

- void mergeParts (int *part1, int len1, int *part2, int len2, int *dest)

  Merge two arrays in the right way.
- void sortRecursive (int *arr, int len)

  The recursive layer for merge sort.

Additional Inherited Members

4.3.1 Detailed Description

MergeSorter class.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 MergeSorter::MergeSorter ( )

Default constructor for Sorters.

Constructs with name: "Merge Sorter"

4.3.3 Member Function Documentation

4.3.3.1 void MergeSorter::mergeParts (int * *part1*, int *len1*, int * *part2*, int *len2*, int * *dest*) [private]

Merge two arrays in the right way.

Parameters

part1	The first part to merge.
len1	The length of the first part.
part2	The second part to merge.
len2	The length of the second part.
dest	The merge destination.

Precondition

The length of dest >= len1+len2

4.3.3.2 void MergeSorter::sortInPlace (int * *arr*, int *len*) [virtual]

Sorts an array in place.

Uses the merge sort algorithm.

Implements Sorter.

4.3.3.3 void MergeSorter::sortRecursive (int * *arr*, int *len*) [private]

The recursive layer for merge sort.

Parameters

arr	The array to sort.
len	The length of the array to sort

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

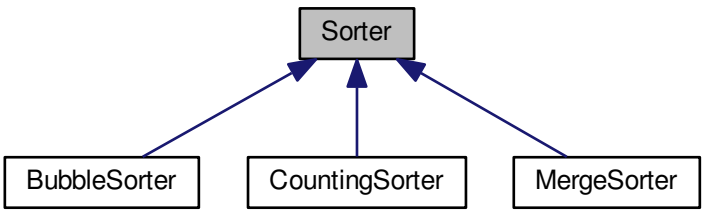
- MergeSorter.h
- MergeSorter.cpp

4.4 Sorter Class Reference

Base class for Sorter classes, which sort int data.

```
#include <Sorter.h>
```

Inheritance diagram for Sorter:



Public Member Functions

- Sorter ()

  Default constructor for Sorters.
- virtual void sortInPlace (int *arr, int len)=0

  Sorts an array in place.
- SortStats getStats ()

  Returns statistics on all sorts performed.
- void resetStats ()

  Resets the stats.
- std::string getName ()

  Gets the name of this sorter.

Protected Member Functions

- void startTimer ()

  Starts timing the algorithm.
- void stopTimer ()

  Stops timing the algorithm and adds the elapsed time to totTime.

Protected Attributes

- int sortsRun
- long totTime
- long totNumComparisons
- long totNumSwaps
- std::string name
- std::chrono::time_point< std::chrono::high_resolution_clock > tmrStart

4.4.1 Detailed Description

Base class for Sorter classes, which sort int data.

Sorter classes record various statistics on the sort.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 Sorter::Sorter ( )

Default constructor for Sorters.

Constructs with name: "Base Sorter"

4.4.3 Member Function Documentation

4.4.3.1 std::string Sorter::getName ( )

Gets the name of this sorter.

Returns

The name of this sorter.

4.4.3.2 SortStats Sorter::getStats ( )

Returns statistics on all sorts performed.

Returns

A filled SortStats instance.

4.4.3.3 void Sorter::resetStats ( )

Resets the stats.

4.4.3.4 virtual void Sorter::sortInPlace (int * *arr*, int *len*) [pure virtual]

Sorts an array in place.

Parameters

arr	The array to sort.
len	The length of the array to sort.

Precondition

The given array is at least len long.

Algorithm is abstracted.

Implemented in MergeSorter, BubbleSorter, and CountingSorter.

4.4.3.5 void Sorter::startTimer ( ) [protected]

Starts timing the algorithm.

Precondition

The timer is stopped.

4.4.3.6 void Sorter::stopTimer ( ) [protected]

Stops timing the algorithm and adds the elapsed time to totTime.

Precondition

The timer has been started.

4.4.4 Member Data Documentation

4.4.4.1 std::string Sorter::name [protected]

4.4.4.2 int Sorter::sortsRun [protected]

4.4.4.3 std::chrono::time_point<std::chrono::high_resolution_clock> Sorter::tmrStart [protected]

4.4.4.4 long Sorter::totNumComparisons [protected]

4.4.4.5 long Sorter::totNumSwaps [protected]

4.4.4.6 long Sorter::totTime [protected]

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

- Sorter.h
- Sorter.cpp

4.5 SortStats Struct Reference

Contains sorting statistics.

`#include <Sorter.h>`

Public Attributes

- float avgTime
- float avgNumComparisons
- float avgNumSwaps

4.5.1 Detailed Description

Contains sorting statistics.

4.5.2 Member Data Documentation

4.5.2.1 float SortStats::avgNumComparisons

4.5.2.2 float SortStats::avgNumSwaps

4.5.2.3 float SortStats::avgTime

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

- Sorter.h

Chapter 5

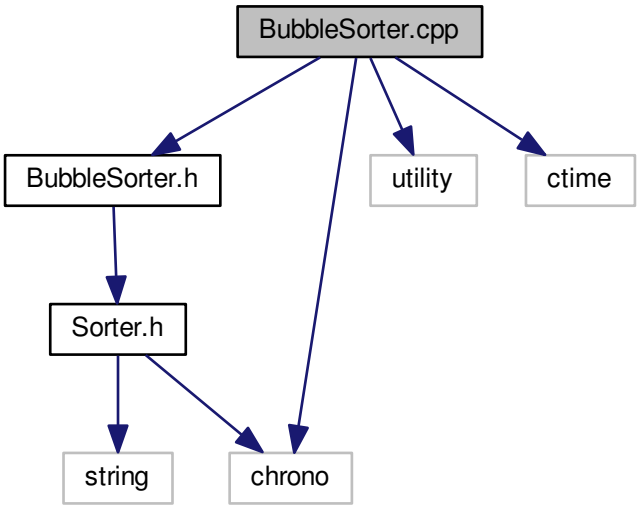
File Documentation

5.1 BubbleSorter.cpp File Reference

Implementation file for the BubbleSorter class.

```
#include "BubbleSorter.h"
#include <utility>
#include <ctime>
#include <chrono>
```

Include dependency graph for BubbleSorter.cpp:



5.1.1 Detailed Description

Implementation file for the BubbleSorter class.

Author

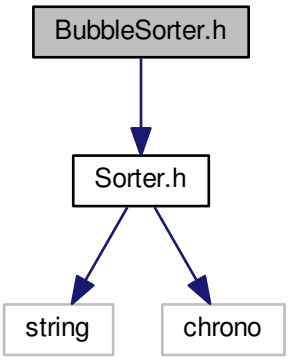
Matthew Bauer

5.2 BubbleSorter.h File Reference

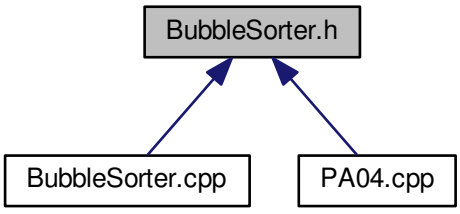
Delcaration file for the BubbleSorter class.

```
#include "Sorter.h"
```

Include dependency graph for BubbleSorter.h:



This graph shows which files directly or indirectly include this file:



Classes

- class BubbleSorter
  BubbleSorter class.

5.2.1 Detailed Description

Delcaration file for the BubbleSorter class.

Author

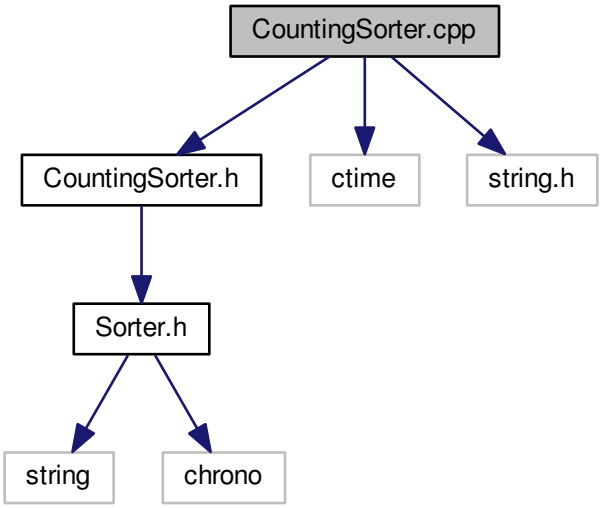
Matthew Bauer

5.3 CountingSorter.cpp File Reference

Implementation file for the CountingSorter class.

```
#include "CountingSorter.h"
#include <ctime>
#include <string.h>
```

Include dependency graph for CountingSorter.cpp:



5.3.1 Detailed Description

Implementation file for the CountingSorter class.

Author

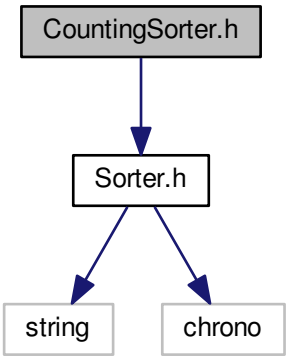
Matthew Bauer

5.4 CountingSorter.h File Reference

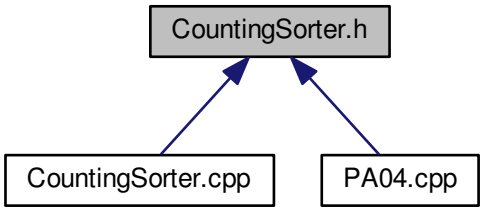
Declaration file for the CountingSorter class.

```
#include "Sorter.h"
```

Include dependency graph for CountingSorter.h:



This graph shows which files directly or indirectly include this file:



Classes

- class CountingSorter

  CountingSorter class.

5.4.1 Detailed Description

Declaration file for the CountingSorter class.

Author

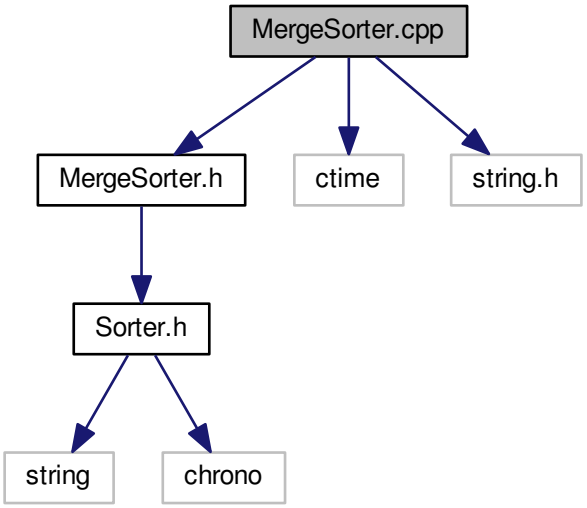
Matthew Bauer

5.5 MergeSorter.cpp File Reference

Implementation file for the MergeSorter class.

```
#include "MergeSorter.h"
#include <ctime>
#include <string.h>
```

Include dependency graph for MergeSorter.cpp:



5.5.1 Detailed Description

Implementation file for the MergeSorter class.

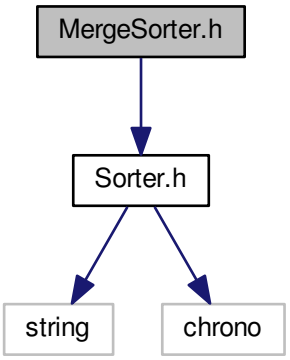
Author

Matthew Bauer

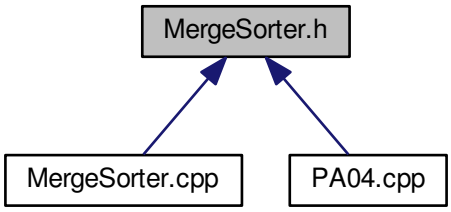
5.6 MergeSorter.h File Reference

Declaration file for the MergeSorter class.

```
#include "Sorter.h"
```
Include dependency graph for MergeSorter.h:



This graph shows which files directly or indirectly include this file:



Classes

- class MergeSorter

  MergeSorter class.

5.6.1 Detailed Description

Declaration file for the MergeSorter class.

Author

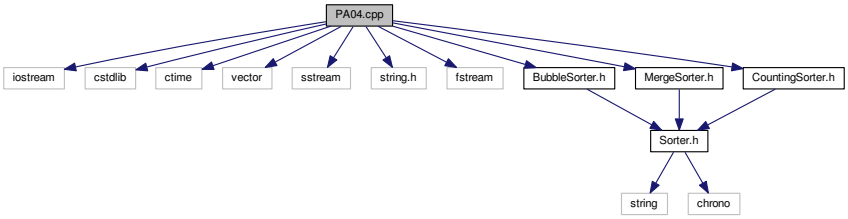
Matthew Bauer

5.7 PA04.cpp File Reference

Main file for CS302/PA04.

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <vector>
#include <sstream>
#include <string.h>
#include <fstream>
#include "BubbleSorter.h"
#include "MergeSorter.h"
#include "CountingSorter.h"
```

Include dependency graph for PA04.cpp:



Functions

- void fillArrayRandomly (int *arr, int numInts)

  Fills an array with the given number of random integers with values ranging from 0 to 1000000.
- void fillRandset (int **&set, int nval)

  Fill one randset.
- void generateRandsets ()

  Fill all randsets appropriately.
- void freeOneRandset (int **&set, bool is1m=false)

  Free one randset.
- void freeRandsets ()

  Free the memory of all randsets.
- void testArray (Sorter *sorter, int *arr, int len, std::stringstream &sortOut, bool writeOut=false)

  Test an array on a Sorter.
- void testRandset (Sorter *sorter, int **set, int len, std::stringstream &sortOut, std::stringstream &statOut)

  Test a randset on a Sorter.
- void testOneSorter (Sorter *sorter, std::stringstream &sortOut, std::stringstream &statOut)

  Test a Sorter on various data.
- void testSorters (std::stringstream &sortOut, std::stringstream &statOut)

  Tests all Sorters fully.
- int main ()

  Entry point.

Variables

- int ** randset_1k
- int ** randset_10k
- int ** randset_100k
- int ** randset_1m

5.7.1 Detailed Description

Main file for CS302/PA04.

Author

Matthew Bauer

5.7.2 Function Documentation

5.7.2.1 void fillArrayRandomly (int * *arr,* int *numInts*)

Fills an array with the given number of random integers with values ranging from 0 to 1000000.

Parameters

array	The array to fill.
numInts	The number of ints to fill it with.

Precondition

The given array must be allocated with more than numInts*4 bytes of space.

Postcondition

The first numInts elements of the array will be filled with random integers.

5.7.2.2 void fillRandset (int **& *set,* int *nval*)

Fill one randset.

Parameters

set	The randset to fill.
nval	The N-value of the randset.

Precondition

The given randset has not been allocated.

5.7.2.3 void freeOneRandset (int **& *set*, bool *is1m* = false)

Free one randset.

Parameters

set	The randset to free.
is1m	If the randset is the randset with n=1 million

Precondition

The randset is allocated and valid.

Postcondition

The randset pointer will be set to nullptr.

5.7.2.4 void freeRandsets ( )

Free the memory of all randsets.

Postcondition

All randset pointers will be set to nullptr.

5.7.2.5 void generateRandsets ( )

Fill all randsets appropriately.

5.7.2.6 int main ( )

Entry point.

Returns

0 (assumes no failure)

5.7.2.7 void testArray (Sorter * *sorter*, int * *arr*, int *len*, std::stringstream & *sortOut*, bool *writeOut* = false)

Test an array on a Sorter.

Parameters

sorter	The Sorter to test the array on.
arr	The array to test.
len	The length of the array to test.
sortOut	The output stream to print sort results to.
writeOut	Whether output should be added to the stream.

5.7.2.8 void testOneSorter (Sorter * *sorter,* std::stringstream & *sortOut,* std::stringstream & *statOut*)

Test a Sorter on various data.

Parameters

sorter	The Sorter to test.
sortOut	The output stream to print sort results to.
statOut	The otuput stream to print stats to.

Precondition

Global randsets have been generated.

5.7.2.9 void testRandset (Sorter * *sorter,* int ** *set,* int *len,* std::stringstream & *sortOut,* std::stringstream & *statOut*)

Test a randset on a Sorter.

Parameters

sorter	The Sorter to test the randset on.
set	The randset to test.
len	The length of the arrays in the randset to test.
sortOut	The output stream to print sort results to.
statOut	The output stream to print stats to.

5.7.2.10 void testSorters (std::stringstream & *sortOut,* std::stringstream & *statOut*)

Tests all Sorters fully.

Prints output to files: sorts.txt, stats.txt

5.7.3 Variable Documentation

5.7.3.1 int ** randset_100k

5.7.3.2 int ** randset_10k

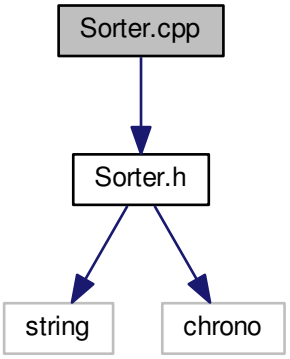
5.7.3.3 int** randset_1k

5.7.3.4 int ** randset_1m

5.8 Sorter.cpp File Reference

Implementation file for basic sorting structures.

```
#include "Sorter.h"
```
Include dependency graph for Sorter.cpp:



5.8.1 Detailed Description

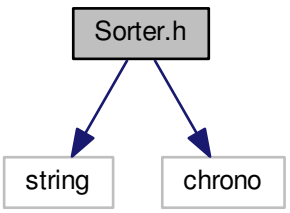
Implementation file for basic sorting structures.

Author

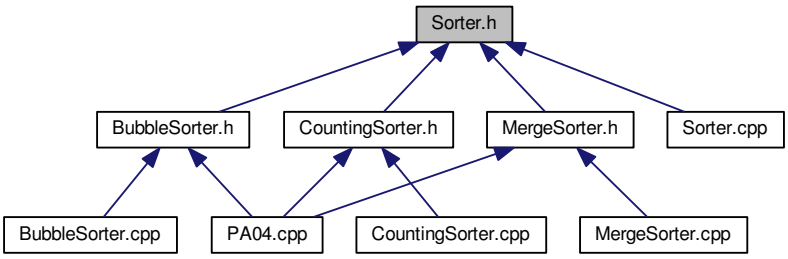
5.9 Sorter.h File Reference

Declaration file for basic sorting structures.

```
#include <string>
#include <chrono>
```
Include dependency graph for Sorter.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct SortStats

  Contains sorting statistics.
- class Sorter

  Base class for Sorter classes, which sort int data.

5.9.1 Detailed Description

Declaration file for basic sorting structures.

Author

Matthew Bauer

Index