AlgorithmAnalysis

1.1

Generated by Doxygen 1.7.5.1

Sun Dec 16 2012 15:23:45

Contents

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

Chapter 2

File Documentation

2.1 include/aux.h File Reference

```
#include <time.h> #include <cstdlib> #include <iostream> ×
#include "../include/ordenacion.h"    #include <fstream> ×
#include <iomanip>
```

Functions

- void createRandomArray (int V[], int arrayLegth, int maxInteger)

  Creates a random integer array between 0 and.
- void copyArray (int V[], int W[], int n)
- bool checkArrayEquality (int V[], int W[], int n)
- bool checkIsOrdered (int V[], int n)
- void printArrayContents (int V[], int n)

2.1.1 Function Documentation

2.1.1.1 bool checkArrayEquality (int *V[],* int *W[],* int *n*)

2.1.1.2 bool checkIsOrdered (int *V[],* int *n*)

2.1.1.3 void copyArray (int *V[],* int *W[],* int *n*)

2.1.1.4 void createRandomArray (int *V[],* int *arrayLegth,* int *maxInteger*)

Creates a random integer array between 0 and.

2.1.1.5 void printArrayContents (int *V[],* int *n*)

2.2 include/generators.h File Reference

```
#include "aux.h"
```

Functions

- void generateInsertionSortTime (ofstream &file, int V[], int n)
- void generateSelectionSortTime (ofstream &file, int V[], int n)
- void generateBubbleSortTime (ofstream &file, int V[], int n)
- void generateQuickSortTime (ofstream &file, int V[], int n)
- void generateInsertionSortDataFile (int problemSize, int V[], int GAP)
- void generateSelectionSortDataFile (int problemSize, int V[], int GAP)
- void generateBubbleSortDataFile (int problemSize, int V[], int GAP)
- void generateQuickSortDataFile (int problemSize, int V[], int GAP)
- void generateAllFiles (int problemSize, int V[], int GAP)

2.2.1 Function Documentation

2.2.1.1 void generateAllFiles (int *problemSize,* int *V[],* int *GAP*)

2.2.1.2 void generateBubbleSortDataFile (int *problemSize,* int *V[],* int *GAP*)

2.2.1.3 void generateBubbleSortTime (ofstream & *file,* int *V[],* int *n*)

2.2.1.4 void generateInsertionSortDataFile (int *problemSize,* int *V[],* int *GAP*)

2.2.1.5 void generateInsertionSortTime (ofstream & *file,* int *V[],* int *n*)

2.2.1.6 void generateQuickSortDataFile (int *problemSize,* int *V[],* int *GAP*)

2.2.1.7 void generateQuickSortTime (ofstream & *file,* int *V[],* int *n*)

2.2.1.8 void generateSelectionSortDataFile (int *problemSize,* int *V[],* int *GAP*)

2.2.1.9 void generateSelectionSortTime (ofstream & *file,* int *V[],* int *n*)

2.3 include/interaction.h File Reference

Functions

- int getProblemSize ()

  Sets problem's size (random array's size)
- int getMaximumInteger ()
- int getGap ()
- char chooseAlgorithm ()

2.3.1 Function Documentation

2.3.1.1 char chooseAlgorithm ( )

2.3.1.2 int getGap ( )

2.3.1.3 int getMaximumInteger ( )

2.3.1.4 int getProblemSize ( )

Sets problem's size (random array's size)

Returns

a positive integer that will represent problem's size hereinafter

2.4 include/ordenacion.h File Reference

Functions

- void insertionSort (int V[], int num)

  Performs an insertion sort algorithm on a vector of positive integers.
- void selectionSort (int V[], int num)

  Performs a selection sort algorithm on a vector of positive integers.
- void bubbleSort (int V[], int num)

  Performs a bubble sort algorithm on a vector of positive integers.
- void quickSort (int V[], int left, int right)

  Performs a quick sort algorithm on a vector of positive integers.

2.4.1 Function Documentation

2.4.1.1 void bubbleSort (int *V[],* int *num*)

Performs a bubble sort algorithm on a vector of positive integers.

Parameters

V	a vector of positive integers
num	The array length

2.4.1.2 void insertionSort (int *V[],* int *num*)

Performs an insertion sort algorithm on a vector of positive integers.

Parameters

V	a vector of positive integers
num	The array length

2.4.1.3 void quickSort (int *V[],* int *left,* int *right*)

Performs a quick sort algorithm on a vector of positive integers.

Parameters

V	a vector of positive integers
left	the left index for the divide and conquer strategy (initially 0)
right	the right index for the divide and conquer strategy (initially the array length)

2.4.1.4 void selectionSort (int *V[],* int *num*)

Performs a selection sort algorithm on a vector of positive integers.

Parameters

V	a vector of positive integers
num	The array length

2.5 include/test.h File Reference

```
#include "../include/aux.h"
```

Functions

- void generateArray (int V[], int n, int maxInt)

  Generates an array with n random integers betwwen 0 and maxInt.
- void makeFourCopies (int V[], int first[], int second[], int third[], int fourth[], int n)

2.5.1 Function Documentation

2.5.1.1 void generateArray (int *V[],* int *n,* int *maxInt*)

Generates an array with n random integers betwwen 0 and maxInt.

Parameters

V	the container array with a size of n
n	The array length
maxInt	maximum integer to generate arraty to

2.5.1.2 void makeFourCopies (int *V[],* int *first[],* int *second[],* int *third[],* int *fourth[],* int *n*)

2.6 src/aux.cpp File Reference

```
#include "../include/aux.h"
```

Functions

- void createRandomArray (int V[], int n, int maxInteger)

  Creates a random integer array between 0 and.
- void copyArray (int V[], int W[], int n)
- bool checkSameLength (int V[], int W[])
- bool checkArrayEquality (int V[], int W[], int n)
- bool checkIsOrdered (int V[], int n)
- void printArrayContents (int V[], int n)

2.6.1 Function Documentation

2.6.1.1 bool checkArrayEquality (int *V[],* int *W[],* int *n*)

2.6.1.2 bool checkIsOrdered (int *V[],* int *n*)

2.6.1.3 bool checkSameLength (int *V[],* int *W[]*)

2.6.1.4 void copyArray (int *V[],* int *W[],* int *n*)

2.6.1.5 void createRandomArray (int *V[],* int *n,* int *maxInteger*)

Creates a random integer array between 0 and.

2.6.1.6 void printArrayContents (int *V[],* int *n*)

2.7 src/generators.cpp File Reference

```
#include "../include/generators.h"
```

Functions

- void generateInsertionSortTime (ofstream &file, int V[], int n)
- void generateSelectionSortTime (ofstream &file, int V[], int n)
- void generateBubbleSortTime (ofstream &file, int V[], int n)
- void generateQuickSortTime (ofstream &file, int V[], int n)
- void generateInsertionSortDataFile (int problemSize, int V[], int GAP)
- void generateSelectionSortDataFile (int problemSize, int V[], int GAP)
- void generateBubbleSortDataFile (int problemSize, int V[], int GAP)
- void generateQuickSortDataFile (int problemSize, int V[], int GAP)
- void generateAllFiles (int problemSize, int V[], int GAP)

2.7.1 Function Documentation

2.7.1.1 void generateAllFiles (int *problemSize,* int *V[],* int *GAP*)

2.7.1.2 void generateBubbleSortDataFile (int *problemSize,* int *V[],* int *GAP*)

2.7.1.3 void generateBubbleSortTime (ofstream & *file,* int *V[],* int *n*)

2.7.1.4 void generateInsertionSortDataFile (int *problemSize,* int *V[],* int *GAP*)

2.7.1.5 void generateInsertionSortTime (ofstream & *file,* int *V[],* int *n*)

2.7.1.6 void generateQuickSortDataFile (int *problemSize,* int *V[],* int *GAP*)

2.7.1.7 void generateQuickSortTime (ofstream & *file,* int *V[],* int *n*)

2.7.1.8 void generateSelectionSortDataFile (int *problemSize,* int *V[],* int *GAP*)

2.7.1.9 void generateSelectionSortTime (ofstream & *file,* int *V[],* int *n*)

2.8 src/interaction.cpp File Reference

```
#include "../include/interaction.h" #include <iostream>
```

Functions

- int getProblemSize ()

  Sets problem's size (random array's size)
- int getMaximumInteger ()
- int getGap ()
- char chooseAlgorithm ()

2.8.1 Function Documentation

2.8.1.1 char chooseAlgorithm ( )

2.8.1.2 int getGap ( )

2.8.1.3 int getMaximumInteger ( )

2.8.1.4 int getProblemSize ( )

Sets problem's size (random array's size)

Returns

a positive integer that will represent problem's size hereinafter

2.9 src/main.cpp File Reference

```
#include "../include/test.h" #include "../include/interaction.-
h" #include "../include/generators.h"
```

Functions

- int main ()

2.9.1 Function Documentation

2.9.1.1 int main ( )

2.10 src/ordenacion.cpp File Reference

```
#include "../include/ordenacion.h"
```

Functions

- void insertionSort (int V[], int num)

  Performs an insertion sort algorithm on a vector of positive integers.
- void selectionSort (int V[], int num)

  Performs a selection sort algorithm on a vector of positive integers.
- void bubbleSort (int V[], int num)

  Performs a bubble sort algorithm on a vector of positive integers.
- void quickSort (int V[], int left, int right)

  Performs a quick sort algorithm on a vector of positive integers.
- void merge (int ∗a, int ∗b, int low, int pivot, int high)

2.10.1 Function Documentation

2.10.1.1 void bubbleSort (int *V[],* int *num*)

Performs a bubble sort algorithm on a vector of positive integers.

Parameters

V	a vector of positive integers
num	The array length

2.10.1.2 void insertionSort (int *V[],* int *num*)

Performs an insertion sort algorithm on a vector of positive integers.

Parameters

V	a vector of positive integers
num	The array length

2.10.1.3 void merge (int * *a,* int * *b,* int *low,* int *pivot,* int *high*)

2.10.1.4 void quickSort (int *V[],* int *left,* int *right*)

Performs a quick sort algorithm on a vector of positive integers.

Parameters

V	a vector of positive integers
left	the left index for the divide and conquer strategy (initially 0)
right	the right index for the divide and conquer strategy (initially the array length)

2.10.1.5 void selectionSort (int *V[],* int *num*)

Performs a selection sort algorithm on a vector of positive integers.

Parameters

V	a vector of positive integers
num	The array length

2.11 src/test.cpp File Reference

```
#include "../include/test.h"
```

Functions

- void makeFourCopies (int V[], int first[], int second[], int third[], int fourth[], int n)
- void generateArray (int V[], int n, int maxInt)

  Generates an array with n random integers betwwen 0 and maxInt.

2.11.1 Function Documentation

2.11.1.1 void generateArray (int *V[],* int *n,* int *maxInt*)

Generates an array with n random integers betwwen 0 and maxInt.

Parameters

V	the container array with a size of n
n	The array length
maxInt	maximum integer to generate arraty to

2.11.1.2 void makeFourCopies (int *V[],* int *first[],* int *second[],* int *third[],* int *fourth[],* int *n*)