Project 1

Generated by Doxygen 1.9.1

1 Class Index 1

1.1 Class List 1

2 File Index 3

2.1 File List 3

3 Class Documentation 5

3.1 Array Class Reference 5

3.1.1 Detailed Description 6

3.1.2 Constructor & Destructor Documentation 6

3.1.2.1 Array() `[1/2]` 6

3.1.2.2 Array() `[2/2]` 7

3.1.2.3 ∼Array() 7

3.1.3 Member Function Documentation 7

3.1.3.1 getArrayCount() 7

3.1.3.2 getSize() 7

3.1.3.3 operator"!=() 8

3.1.3.4 operator=() 8

3.1.3.5 operator==() 8

3.1.3.6 operator[]() 9

3.1.4 Friends And Related Function Documentation 9

3.1.4.1 operator<< 9

3.1.4.2 operator>> 9

3.1.5 Member Data Documentation 10

3.1.5.1 arrayCount 10

3.1.5.2 Author: Deitel/Deitel (Additional comments by Olson and Zander) 10

3.1.5.3 – in <<, integers are displayed 10 per line 10

3.1.5.4 ptr 10

3.1.5.5 size 10

4 File Documentation 11

4.1 1array.cpp File Reference 11

4.1.1 Function Documentation 11

4.1.1.1 operator<<() 11

4.1.1.2 operator>>() 12

4.2 1array.cpp 12

4.3 1array.h File Reference 13

4.4 1array.h 14

Index 15

Chapter 1

Class Index

1.1 Class List

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

Array 5

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

1array.cpp 11
1array.h 13

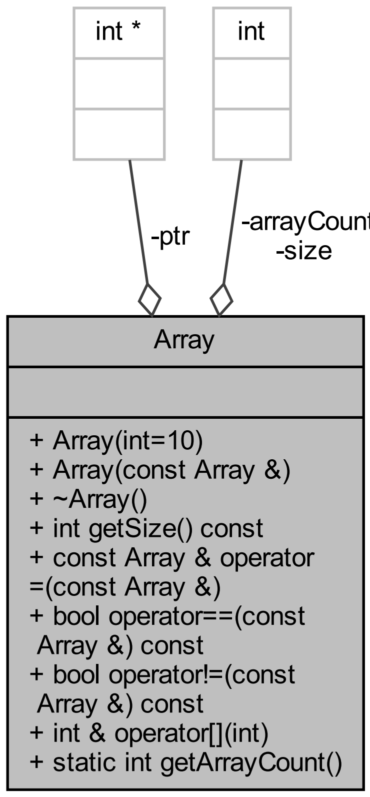
Chapter 3

Class Documentation

3.1 Array Class Reference

```
#include <1array.h>
```

Collaboration diagram for Array:


int *
int
-ptr
-arrayCount
-size
Array
+ Array(int=10)
+ Array(const Array &)
+ ~Array()
+ int getSize() const
+ const Array & operator =(const Array &)
+ bool operator==(const Array &) const
+ bool operator!=(const Array &) const
+ int & operator[](int)
+ static int getArrayCount()

Public Member Functions

- Array (int=10)
- Array (const Array &)
- ∼Array ()
- int getSize () const
- const Array & operator= (const Array &)
- bool operator== (const Array &) const
- bool operator!= (const Array &) const
- int & operator[ ] (int)

Static Public Member Functions

- static int getArrayCount ()

Private Attributes

- int ∗ ptr
- int size

Static Private Attributes

- static int arrayCount = 0

  Initialize static data member at file scope.

Friends

- istream & operator>> (istream &, Array &)
- ostream & operator<< (ostream &, const Array &)

3.1.1 Detailed Description

Definition at line 28 of file 1array.h.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Array() [1/2]

```
Array::Array (
                int arraySize = 10 )
```

Default constructor

Precondition

Preconditions: None

Postcondition

Postconditions: ptr points to an array of size arraySize and all elements of the array have been initialized to zero. arrayCount is incremented Negative input values result in the default size of 10

Definition at line 36 of file 1array.cpp.

3.1.2.2 Array() [2/2]

```
Array::Array (
                    const Array & init )
```

Copy constructor

Precondition

Preconditions: init.ptr points to an array of size at least init.size

Postcondition

Postconditions: init is copied into ∗this, arrayCount is incremented

Definition at line 51 of file 1array.cpp.

3.1.2.3 ∼Array()

```
Array::∼Array ( )
```

Destructor

Precondition

Preconditions: ptr points to memory on the heap

Postcondition

Postconditions: Array for ptr is deallocated, arrayCount is decremented

Definition at line 65 of file 1array.cpp.

3.1.3 Member Function Documentation

3.1.3.1 getArrayCount()

```
int Array::getArrayCount ( ) [static]
```

getArrayCount Return the number of Array objects instantiated

Returns

returns number of arrays in arrayCount

Precondition

Preconditions: None

Postcondition

Postconditions: Returns the number of arrays

Definition at line 145 of file 1array.cpp.

3.1.3.2 getSize()

```
int Array::getSize ( ) const
```

getSize returns the size of the array

Returns

getSize returns the size of the array

Precondition

Preconditions: None

Postcondition

Postconditions: Returns the size of the array

Definition at line 76 of file 1array.cpp.

3.1.3.3 operator"!=()

```
bool Array::operator!= (
                const Array & right ) const
```

operator!= Determine if two arrays are not equal.

Returns

boolean true or false depending on size

Precondition

Preconditions: ptr and right.ptr point to arrays with size at least size and right.size, respectively

Postcondition

Postconditions: false is returned if the arrays have the same size and elements true is return otherwise

Definition at line 124 of file 1array.cpp.

3.1.3.4 operator=()

```
const Array & Array::operator= (
                const Array & right )
```

operator= Overwrites left parameter with right parameter

Returns

ptr of this

Precondition

Preconditions: right.ptr points to an array of size at least right.size

Postcondition

Postconditions: ∗this is assigned the same array as right

Definition at line 84 of file 1array.cpp.

3.1.3.5 operator==()

```
bool Array::operator== (
                const Array & right ) const
```

operator== Determine if two arrays are equal.

Returns

boolean true or false depending on size

Precondition

Preconditions: ptr and right.ptr point to arrays with size at least size and right.size, respectively

Postcondition

Postconditions: true is returned if the arrays have the same size and elements false is return otherwise

Definition at line 106 of file 1array.cpp.

3.1.3.6 operator[]()

```
int & Array::operator[] (
              int subscript )
```

operator[] Overloaded subscript operator, terminates if subscript out of range error

Returns

returns ptr

Precondition

Preconditions: 0 <= subscript < size

Postcondition

Postconditions: Returns the array value at position "subscript"

Definition at line 134 of file 1array.cpp.

3.1.4 Friends And Related Function Documentation

3.1.4.1 operator<<

```
ostream& operator<< (
              ostream & output,
              const Array & a ) [friend]
```

operator<< Overloaded output operator for class Array

Returns

returns output to ostream

Precondition

Preconditions: a.ptr must point to an array with size at least a.size

Postcondition

Postconditions: The first a.size elements of a.ptr are sent to the output istream 10 per line with a trailing endl

Definition at line 168 of file 1array.cpp.

3.1.4.2 operator>>

```
istream& operator>> (
              istream & input,
              Array & a ) [friend]
```

operator>> Overloaded input operator for class Array; inputs values for entire array.

Returns

returns input from istream

Precondition

Preconditions: a.ptr must point to an array with size at least a.size

Postcondition

Postconditions: The first a.size elements of a.ptr are filled with integers read from the input istream

Definition at line 155 of file 1array.cpp.

3.1.5 Member Data Documentation

3.1.5.1 arrayCount

```
int Array::arrayCount = 0  [static], [private]
```

ARRAY.CPP Member function definitions for class Array

3.1.5.2 Author: Deitel/Deitel (Additional comments by Olson and Zander)

Array class: like an int array (retains all functionality) but also includes additional features: – allows input and output of the whole array – allows for comparison of 2 arrays, element by element – allows for assignment of 2 arrays – size is part of the class (so no longer needs to be passed) – includes range checking, program terminates for out-of-bound subscripts
Assumptions: – size defaults to a fixed size of 10 if size is not specified – array elements are initialized to zero – user must enter valid integers when using $>>$

3.1.5.3 – in $<<$, integers are displayed 10 per line

Definition at line 116 of file 1array.h.

3.1.5.4 ptr

```
int* Array::ptr  [private]
```

Definition at line 114 of file 1array.h.

3.1.5.5 size

```
int Array::size  [private]
```

Definition at line 115 of file 1array.h.
The documentation for this class was generated from the following files:

- 1array.h
- 1array.cpp

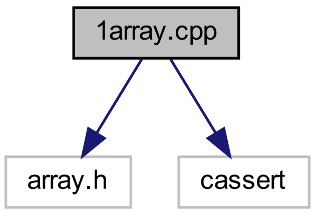
Chapter 4

File Documentation

4.1 1array.cpp File Reference

```
#include "array.h"
#include <cassert>
```

Include dependency graph for 1array.cpp:



Functions

- istream & operator>> (istream &input, Array &a)
- ostream & operator<< (ostream &output, const Array &a)

4.1.1 Function Documentation

4.1.1.1 operator<<()

```
ostream& operator<< (
            ostream & output,
            const Array & a )
```

operator<< Overloaded output operator for class Array

Returns

returns output to ostream

Precondition

Preconditions: a.ptr must point to an array with size at least a.size

Postcondition

Postconditions: The first a.size elements of a.ptr are sent to the output istream 10 per line with a trailing endl

Definition at line 168 of file 1array.cpp.

4.1.1.2 operator>>()

```
istream& operator>> (
              istream & input,
              Array & a )
```

operator>> Overloaded input operator for class Array; inputs values for entire array.

Returns

returns input from istream

Precondition

Preconditions: a.ptr must point to an array with size at least a.size

Postcondition

Postconditions: The first a.size elements of a.ptr are filled with integers read from the input istream

Definition at line 155 of file 1array.cpp.

4.2 1array.cpp

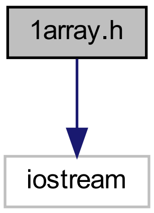
```
00001
00021
00022 #include "array.h"
00023 #include <cassert>
00024
00026 int Array::arrayCount = 0;
00027
00028
00036 Array::Array(int arraySize) {
00037    ++arrayCount;
00038    size = (arraySize > 0 ? arraySize : 10);
00039    ptr = new int[size];
00040    assert(ptr != NULL);
00041
00042    for (int i = 0; i < size; i++)
00043       ptr[i] = 0;
00044 }
00045
00046
00051 Array::Array(const Array &init) {
00052    ++arrayCount;
00053    size = init.size;
00054    ptr = new int[size];
00055    assert(ptr != NULL);
00056
00057    for (int i = 0; i < size; i++)
00058       ptr[i] = init.ptr[i];
00059 }
00060
00065 Array::~Array() {
00066    --arrayCount;
00067    delete [] ptr;
00068 }
00069
00070
00076 int Array::getSize() const { return size; }
00077
00078
00084 const Array& Array::operator=(const Array& right) {
00085    if (&right != this) {
00086       delete [] ptr;
00087       size = right.size;
00088       ptr = new int[size];
00089       assert(ptr != NULL);
00090
00091       for (int i = 0; i < size; i++)
```

```
         ptr[i] = right.ptr[i];
   }

   return *this;
}


bool Array::operator==(const Array& right) const {
   if (size != right.size)
      return false;

   for (int i = 0; i < size; i++)
      if (ptr[i] != right.ptr[i])
         return false;
   return true;
}


bool Array::operator!=(const Array& right) const {
   return !(*this == right);
}


int& Array::operator[](int subscript) {
   assert(0 <= subscript && subscript < size);
   return ptr[subscript];
}


int Array::getArrayCount() { return arrayCount; }



istream& operator»(istream &input, Array &a) {
   for (int i = 0; i < a.size; i++)
      input » a.ptr[i];
   return input;
}


ostream& operator«(ostream &output, const Array &a) {
   int i;
   for (i = 0; i < a.size; i++) {
      output « a.ptr[i] « ' ';
      if ((i + 1) % 10 == 0)
         output « endl;
   }

   if (i % 10 != 0)
      output « endl;
   return output;
}

```

4.3 1array.h File Reference

```
#include <iostream>
```

Include dependency graph for 1array.h:



Classes

- class Array

4.4 1array.h

```

#ifndef ARRAY_H
#define ARRAY_H

#include <iostream>
using namespace std;


class Array {
friend istream& operator»(istream &, Array &);

friend ostream& operator«(ostream &, const Array &);

public:
Array(int = 10);

Array(const Array &);

~Array();

int getSize() const;

const Array& operator=(const Array &);

bool operator==(const Array &) const;

bool operator!=(const Array &) const;

int& operator[](int);

static int getArrayCount();

private:
   int* ptr;                          // pointer to first element of array
   int size;                          // size of the array
   static int arrayCount;             // # of Arrays instantiated
};

#endif

```

Index

~Array
    Array, 7
1array.cpp, 11
    operator<<, 11
    operator>>, 12
1array.h, 13

Array, 5
    ~Array, 7
    Array, 6
    arrayCount, 10
    getArrayCount, 7
    getSize, 7
    operator!=, 8
    operator<<, 9
    operator>>, 9
    operator=, 8
    operator==, 8
    operator[], 8
    ptr, 10
    size, 10
arrayCount
    Array, 10

getArrayCount
    Array, 7
getSize
    Array, 7

operator!=
    Array, 8
operator<<
    1array.cpp, 11
    Array, 9
operator>>
    1array.cpp, 12
    Array, 9
operator=
    Array, 8
operator==
    Array, 8
operator[]
    Array, 8

ptr
    Array, 10

size
    Array, 10