Project4

Generated by Doxygen 1.8.14

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

Class Documentation

2.1 proj4 Class Reference

Public Member Functions

- proj4 ()
- bool insert (int position, int value)
- void clear ()
- void setMaxItems (int items)
- unsigned int getSwaps ()

  returns the amount of swaps
- unsigned int getComparisons ()

  returns the amount of comparisions
- unsigned int getPartitions ()
- unsigned int getCountSorts ()
- unsigned int getCounts ()
- int getReursion ()
- int getLeft ()
- int getRight ()
- int getNumber (int position)

  just a function that shows how it is working
- void bubble_sort ()

  uses two for loops to go through the array and swap neighboring elements
- void swap (int ∗xp, int ∗yp)
- void quick_sort (int low, int high)

  gets the pivot position and uses recursive call to fully sort it
- int partition (int low, int high)

  Will take the pivot positions and compare it to the pivot point to find the postion while swapping.
- void count_sort (int n, int exp)

  has temp array to store the values, then a count array to the position of the number, it then builds the output array and sets the array to the output array
- int getMax (int n)
- void radixsort (int n)
- void merge (int l, int m, int r)

  Will make temp arrays that store the left and right values, then it will merge it back together.
- void mergeSort (int l, int r)

  Will take the left and right to sort the array.

2.1.1 Constructor & Destructor Documentation

2.1.1.1 proj4()

```
proj4::proj4 ( )
```

Will give default values

Precondition

Postcondition

Will give default values

2.1.2 Member Function Documentation

2.1.2.1 bubble_sort()

```
void proj4::bubble_sort ( )
```

uses two for loops to go through the array and swap neighboring elements

Returns

none

Precondition

will sort the array

Postcondition

depending on the size of the array, it will run through it that many times to swap items

2.1.2.2 clear()

```
void proj4::clear ( )
```

Will reset all the values

Precondition

Postcondition

Resets all the values

2.1.2.3 count_sort()

```
void proj4::count_sort (
             int n,
             int exp )
```

has temp array to store the values, then a count array to the position of the number, it then builds the output array and sets the array to the output array

Parameters

n,length	of the array
exp,exponent	it is on

Returns

none

Precondition

takes in the lenght of the array and the exponent it is on to sort the array

Postcondition

takes the exponents and sorts them based on that

2.1.2.4 getComparisons()

```
unsigned int proj4::getComparisons ( )
```

returns the amount of comparisions

Returns

comparisons

2.1.2.5 getCounts()

```
unsigned int proj4::getCounts ( )
```

Will return the amount of times count was used

Returns

counts

Precondition

Postcondition

Will return how many times a values was stored in the array count

2.1.2.6 getCountSorts()

```
unsigned int proj4::getCountSorts ( )
```

Will return how many times sort was called

Returns

countSorts

Precondition

Postcondition

Returns the number of times count sort was called for radix sort

2.1.2.7 getLeft()

```
int proj4::getLeft ( )
```

Will return the amount of items in the left

Returns

left

Precondition

Postcondition

Will return how many times a values was stored in the left array

2.1.2.8 getMax()

```
int proj4::getMax (
             int n )
```

gets the biggest element in the array

Parameters

n	

Returns

temp, largest element

Precondition

takes the length of the array and gets the largest value

Postcondition

runs through the array to find the largest value

2.1.2.9 getNumber()

```
int proj4::getNumber (
            int position )
```

just a function that shows how it is working

Will return the element at that postion.

Parameters

position	

Returns

the element in that array

Precondition

takes in the position to find that element in that postion

Postcondition

Will return the item in that position in the array

2.1.2.10 getPartitions()

```
unsigned int proj4::getPartitions ( )
```

Will return the number of partitions

Returns

Precondition

Postcondition

Returns the number of partitions

2.1.2.11 getReursion()

```
int proj4::getReursion ( )
```

Will return the amount of times recursion was called

Returns

recursions

Precondition

Postcondition

Will return how many times it called itself to sort

2.1.2.12 getRight()

```
int proj4::getRight ( )
```

Will return the amount of items in the right

Returns

right

Precondition

Postcondition

Will return how many times a values was stored in the right array

2.1.2.13 getSwaps()

```
unsigned int proj4::getSwaps ( )
```

returns the amount of swaps

Returns

swaps

2.1.2.14 insert()

```
bool proj4::insert (
             int position,
             int value )
```

Inserts the element in that postion

Parameters

position	
value	

Returns

true or false depending on if it can insert

Precondition

will take the position and value to add to the array

Postcondition

Will take position to give the value to that postion in the array

2.1.2.15 merge()

```
void proj4::merge (
             int l,
             int m,
             int r )
```

Will make temp arrays that store the left and right values, then it will merge it back together.

Parameters

l	
m	
r	

Returns

none

Precondition

Takes in the left right and middle to give sizes to the arrays

Postcondition

Will create temp arrays for the left and right. It will copy the data into both and merge it back to the original, then move any remaining elements back into the original array.

2.1.2.16 mergeSort()

```
void proj4::mergeSort (
             int l,
             int r )
```

Will take the left and right to sort the array.

Parameters

l	
r	

Precondition

Will take in the left and right values to give to the sort

Postcondition

Will use the left and right, and keep using recursive calls to change the left and right values to give to the merge function

2.1.2.17 partition()

```
int proj4::partition (
             int low,
             int high )
```

Will take the pivot positions and compare it to the pivot point to find the postion while swapping.

Parameters

low	
high	

Returns

pivot position

Precondition

will take in the low and high to compare the elemments in the array

Postcondition

will find the find the pivot and find if the element is samller to the pivot and swap them

2.1.2.18 quick_sort()

```
void proj4::quick_sort (
             int low,
             int high )
```

gets the pivot position and uses recursive call to fully sort it

Parameters

low	
high	

Precondition

takes the low and high to find the partition pivot

Postcondition

uses recursion to keep using the partition function to swap the elements till it is sorted

2.1.2.19 radixsort()

```
void proj4::radixsort (
             int n )
```

Will do a count sort for all digits, it uses the exponent to tell what number it is on

Parameters

n	

Returns

none

Precondition

Will take in the length of the array to find the max number, and then call count sort for all the digits

Postcondition

Calls count sort to sort the array

2.1.2.20 setMaxItems()

```
void proj4::setMaxItems (
             int items )
```

gives the max items and amount

Parameters

items	

2.1.2.21 swap()

```
void proj4::swap (
             int * xp,
             int * yp )
```

takes in two values and swaps them

Parameters

xp	
yp	

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

- proj4.h
- proj4.cpp

Index