20161795 고가해

The Sort reports.

RECT DATA SCREEN

```
for (int i = 0; i < MAX_SIZE; i++) {
printf("kuangdu:%f, gaodu:%f didian:%f\r\n", (shuzu + i)->width, (shuzu + i)->gaodu,(shuzu+ i)->kuandu* (shuzu+ i)->gaodu);
   }
```

Conclusion

QUICK SORT

It‘s used divide and rule make the number should sort divided right and left

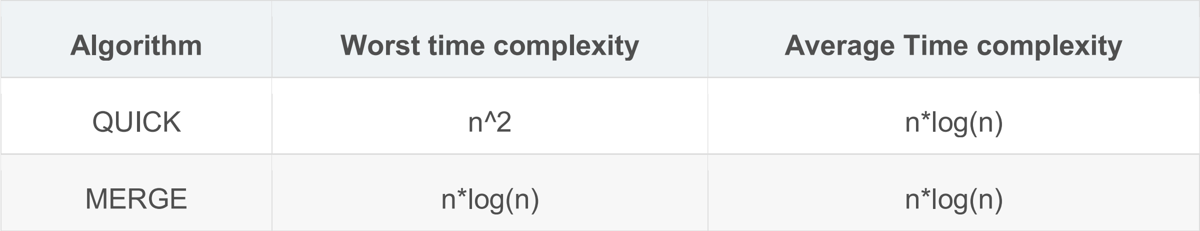
and part of data less than key and the other part is more than key and repeat this operation .the quick sort is quickly but it’s not very stabilization. There’s many error when I was tested

INSERT SORT

Take the element to sorted array, in my sense the insert sort is most stabilization in all sort method

MERGE SORT

This sort method is same as quick sort but if in same environmental it’s not easily to see what’s different when the number is tiny but if the number become large it will obvious to seem the MERGE SORT is good than QUICK SORT



Algorithm	Worst time complexity	Average Time complexity
QUICK	n^2	n*log(n)
MERGE	n*log(n)	n*log(n)

Heap SORT

Max HEAP: arr[i] >= arr[2i+1] && arr[i] >= arr[2i+2]

MIN HEAP: arr[i] <= arr[2i+1] && arr[i] <= arr[2i+2]

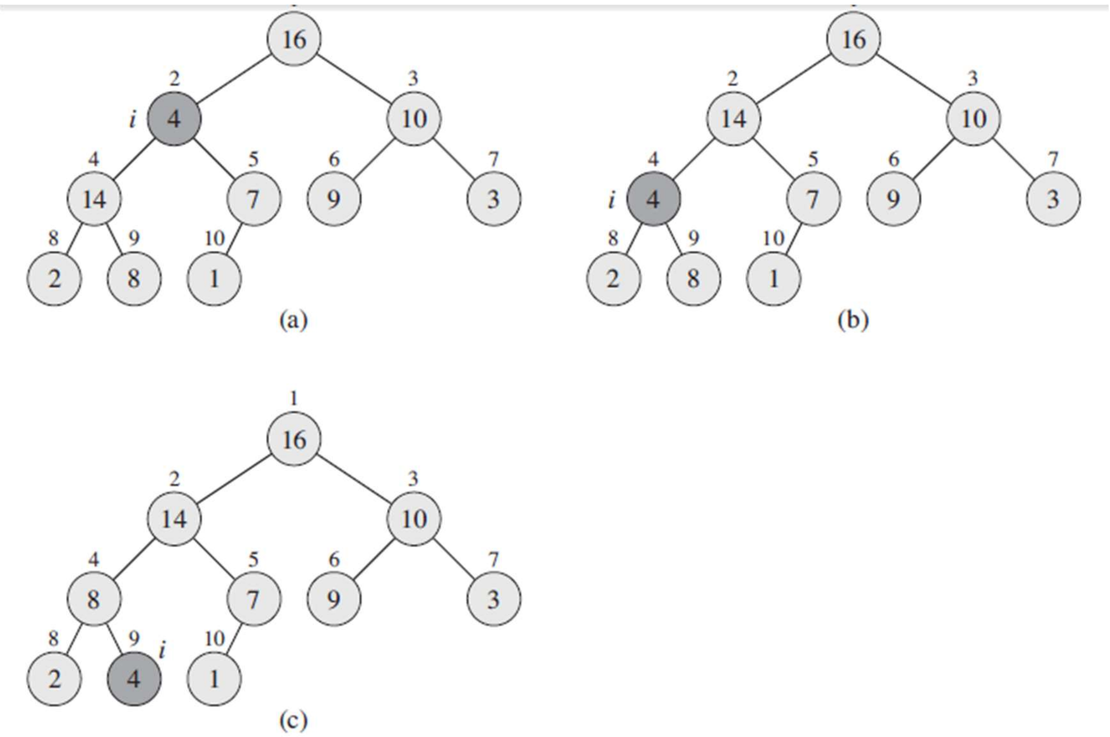
The sequence will be built into a heap, and the large top dump or small top dump will be selected according to the ascending sort descending demand

Swapping the top element of the heap with the last element, "sinking" the largest element to the end of the array

The structure is restructured to satisfy the definition of the heap, and then continue to swap the top element with the current end element, and repeat the adjustment + exchange steps until the entire sequence is in order.

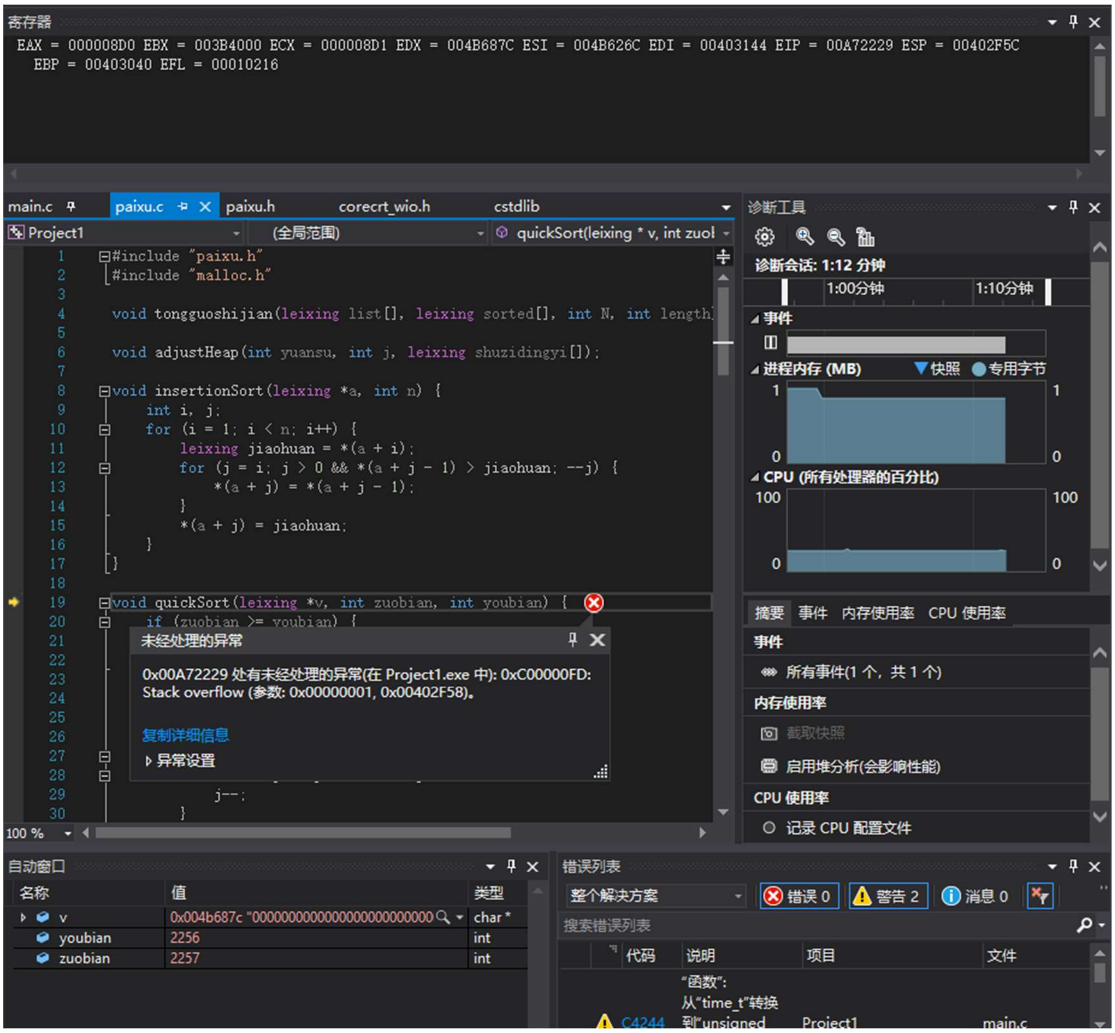
Realize method

4 method and the idea is come from the sort method



(a) (b)

(c)



```
EAX = 000008D0 EBX = 003B4000 ECX = 000008D1 EDX = 004B687C ESI = 004B626C EDI = 00403144 EIP = 00A72229 ESP = 00402F5C
  EBP = 00403040 EFL = 00010216
```

```
#include "paixu.h"
#include "malloc.h"

void tongguoshijian(leixing list[], leixing sorted[], int N, int length]

void adjustHeap(int yuansu, int j, leixing shuzidingyi[]);

void insertionSort(leixing *a, int n) {
    int i, j;
    for (i = 1; i < n; i++) {
        leixing jiaohuan = *(a + i);
        for (j = i; j > 0 && *(a + j - 1) > jiaohuan; --j) {
            *(a + j) = *(a + j - 1);
        }
        *(a + j) = jiaohuan;
    }
}

void quickSort(leixing *v, int zuobian, int youbian) {
    if (zuobian >= youbian) {
```

```
                j--;
            }
```

未经处理的异常

0x00A72229 处有未经处理的异常(在 Project1.exe 中): 0xC00000FD: Stack overflow (参数: 0x00000001, 0x00402F58)。

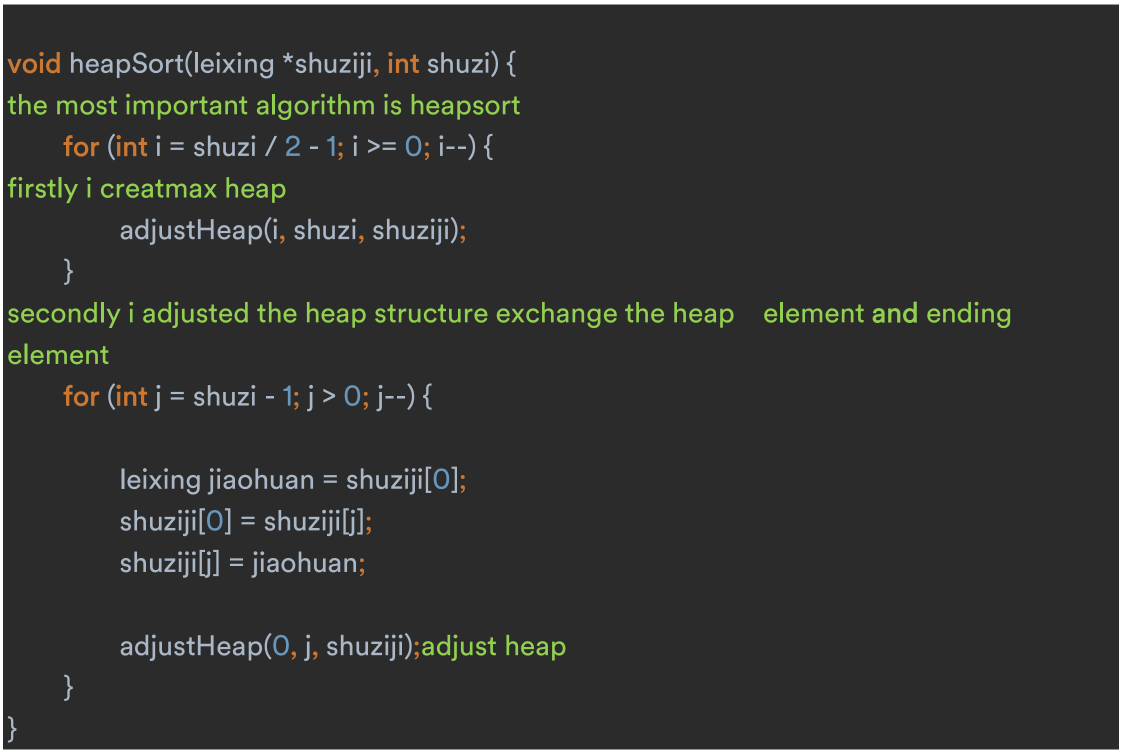
复制详细信息

▷ 异常设置

名称	值	类型
v	0x004b687c "0000000000000000000000000	char *
youbian	2256	int
zuobian	2257	int

代码	说明	项目	文件
C4244	"函数": 从"time_t"转换到"unsigned	Project1	main.c

I used malloc to divide function and made the sort ,but if the data is too large the program will be error in this test and open register and assembly program.

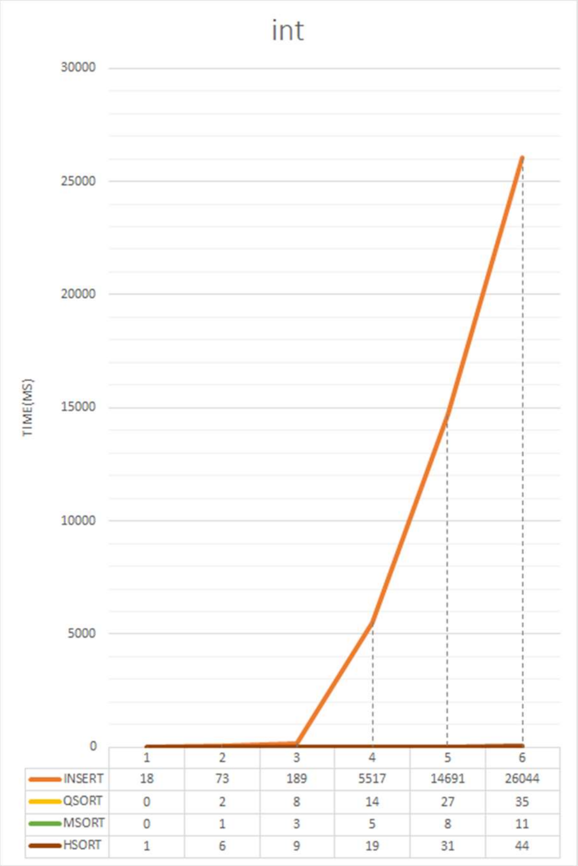


```
void heapSort(leixing *shuziji, int shuzi) {
the most important algorithm is heapsort
    for (int i = shuzi / 2 - 1; i >= 0; i--) {
firstly i creatmax heap
        adjustHeap(i, shuzi, shuziji);
    }
secondly i adjusted the heap structure exchange the heap    element and ending
element
    for (int j = shuzi - 1; j > 0; j--) {

        leixing jiaohuan = shuziji[0];
        shuziji[0] = shuziji[j];
        shuziji[j] = jiaohuan;

        adjustHeap(0, j, shuziji);adjust heap
    }
}
```

The result



	1	2	3	4	5	6
INSERT	18	73	189	5517	14691	26044
QSORT	0	2	8	14	27	35
MSORT	0	1	3	5	8	11
HSORT	1	6	9	19	31	44

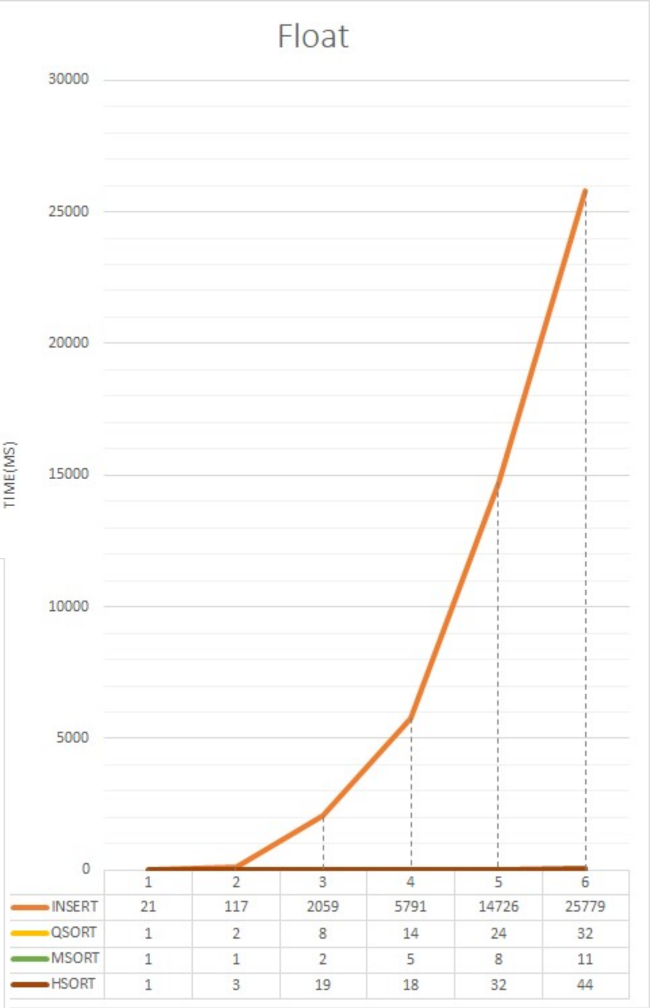
int testing function
(void zhengshu(leixing * shuju);

And you should change the function in the head file

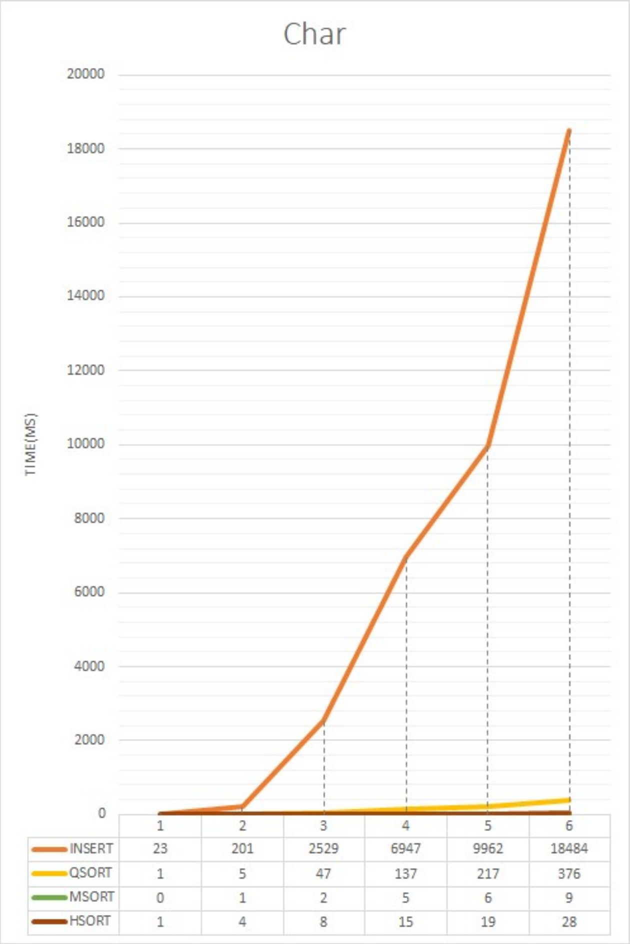
Float

void fudian(leixing * shuju)

And you should change the function in the head file



	1	2	3	4	5	6
INSERT	21	117	2059	5791	14726	25779
QSORT	1	2	8	14	24	32
MSORT	1	1	2	5	8	11
HSORT	1	3	19	18	32	44



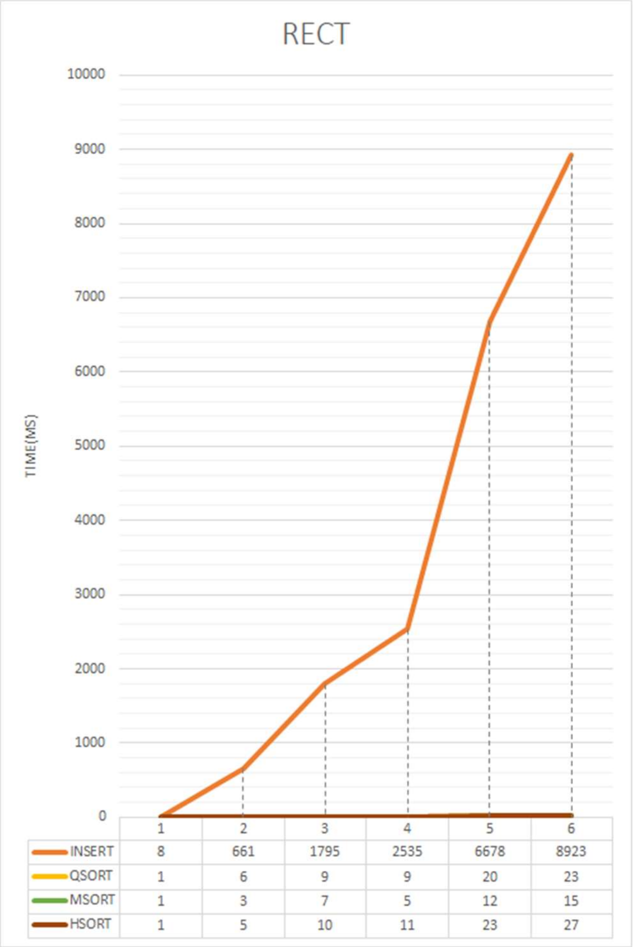
	1	2	3	4	5	6
INSERT	23	201	2529	6947	9962	18484
QSORT	1	5	47	137	217	376
MSORT	0	1	2	5	6	9
HSORT	1	4	8	15	19	28

Character’s testing data is over the standard the asm file has the error so I changed testing data

void zifu(leixing * shuju)

And you should change the function in the head file

RECT’s testing code is also being too large so I changed the testing data and also used insert/quick/merge/heap sort



	1	2	3	4	5	6
INSERT	8	661	1795	2535	6678	8923
QSORT	1	6	9	9	20	23
MSORT	1	3	7	5	12	15
HSORT	1	5	10	11	23	27

Reference data

https://blog.csdn.net/shaya118/article/details/41053833
http://blog.chinaunix.net/uid-25933104-id-3538914.html
https://blog.csdn.net/lgg201/article/details/5018463
https://blog.csdn.net/gl486546/article/details/54744232
https://blog.csdn.net/iluckyning/article/details/8140959
https://blog.csdn.net/gl486546/article/details/54707307
https://blog.csdn.net/pegasuswang_/article/details/20702919