# QG训练营第4周排序算法作业测试报告

## 测试环境及测试工程

### 测试环境

我们使用Visual Studio 2017 作为编译器，本地Windows调试器 win32生成模式，使用默认预编译头

### 测试工程

使用C++作为唯一工程语言

**头文件：**

**Stdafx.h：**默认预编译头文件

**Main.h：**

声明了main.cpp中针对用户界面的封装函数以及其他一些工具函数，3个全局数组指针变量，3个1w 5w 20w的宏

**targetver.h：**忽略

**qgsort.h:**

1. 声明了我们测试使用的各种排序算法函数，统一传参列表

(int\*arry,int length)或(int \*\*arry,int length)

部分排序算法有特殊重载函数

（计数排序：void CountSort(int \*\* arry, int length, int min, int max)）

1. 声明了两个排序应用函数：void ColorSort(int \*arry,int length)彩虹排序和

int kFindSort(int \*arry, int length, int k,bool flag)从大到小/小到大查找无序数组第k个元素的

**源文件：**

**QGHomeWork\_Week4\_AnXiang.cpp：**

定义main函数，实现用户界面封装及其他工具函数，完成测试工程的主流程控制

**qgsort.cpp：**

实现各种排序算法函数，qgsort.cpp中有部分算法的重载函数是作为真正的排序算法函数，未在qgsort.h中定义。qgsort.h 中的重载定义了是为方便使用，统一传参列表的接收函数

**stdafx.cpp:** 忽略

## 测试工程工作流程

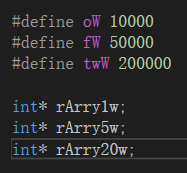
### 1.0主工作流程

我们通过常用的一套while(getcode()); 在getcode()函数中输入命令判定执行，执行后返回循环布尔值作为测试工程的主工作流程

### 1.1排序测试工作流程

**本次测试各排序均采用从小向大排序**

我们在Main.h中声明了分别对应1w 5w 20w数组的指针和宏



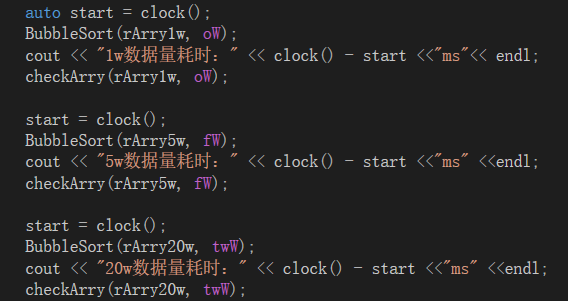
在排序算法开始前我们调用bool setRandArry()结合int \* getRandArry(int length)来完成对数组的初始化

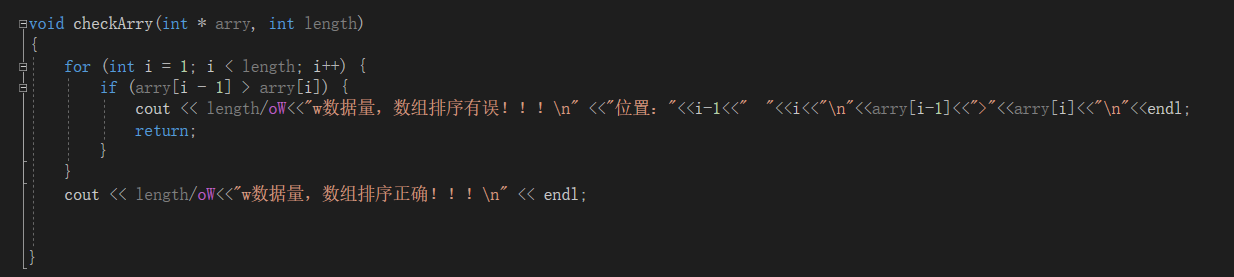
随机数生成我们使用(int)time(0)作为数种，直接使用rand() 的返回结果作为随机数



完成初始化后，我们依次进行 1w 5w 20w的排序

每次排序开始前我们记录 clock() 的返回值，与计数后做比较打印排序耗时（ms）



每次排序结束后我们调用checkArry() 遍历数组来检查排序结果

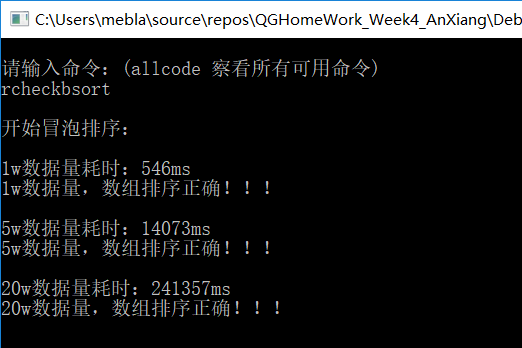
之后释放数组空间，等待下次命令的输入

## 2.测试结果

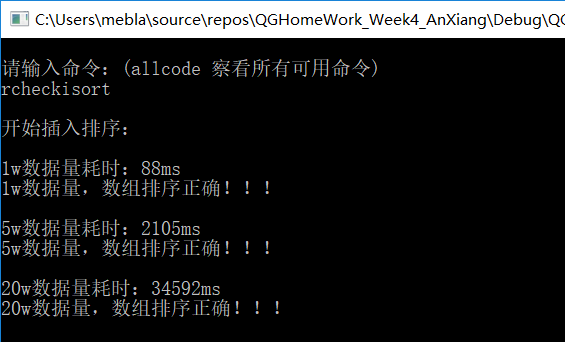
以下是各大排序的测试结果打印

### 冒泡排序

惨遭各方吊打，恐成最大输家

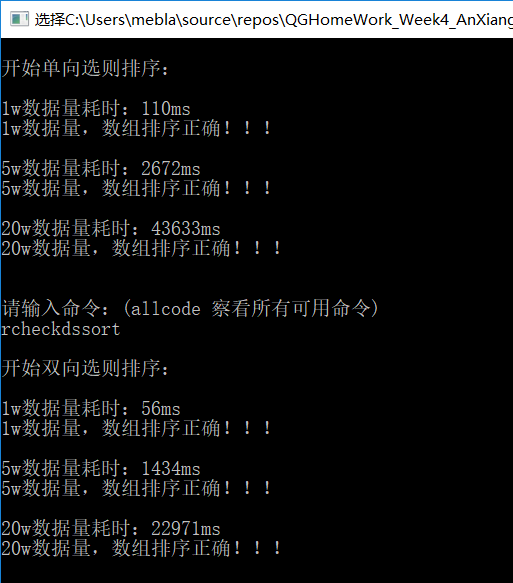


### 插入排序



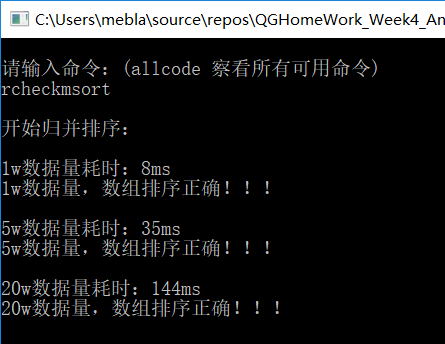
### 单向选则与双向选则排序

可见双向选则对弟弟单向选则优化不少



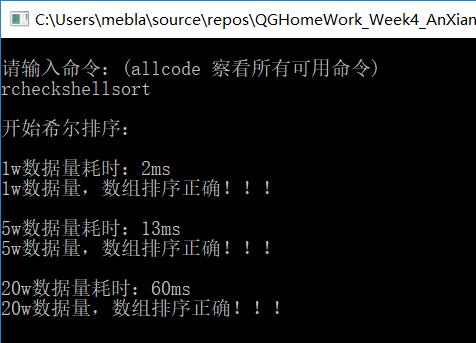
### 归并排序

最佳排序，种子选手之一

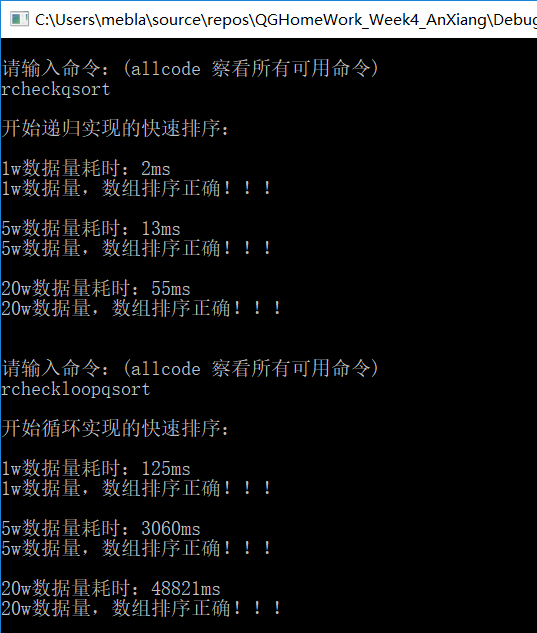


### 希尔排序

上锤归并，硬刚快排，代码易写，不用递归千兆深，不用看脸抽中轴，希尔NB！！！

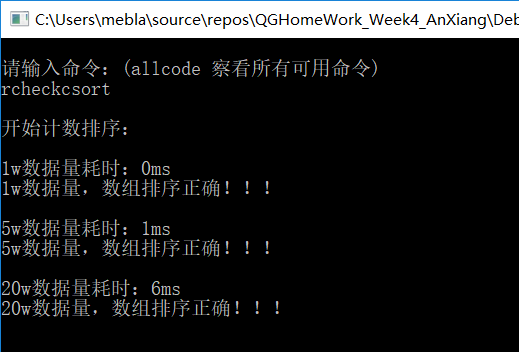


### 递归快排与循环快排



### 计数排序

去tm的内存！！！邪教算法以最短耗时傲视群雄



### 基数排序

