实验报告

一、算法设计思路

1. 快速排序算法实现

首先选择一个元素作为基准元素（pivot），然后通过一趟排序（partition）将整个数组分割成以基准元素为界的两部分，其中一部分的元素数值均小于基准元素，另一部分的元素数值均大于基准元素。然后，可以分别对这两部分重复上述操作，继续进行排序，并重复直至整个数组有序。

其中，在进行 partition 排序时，设置指示位置的变量 i，j，从头开始扫描数据，通过数据之间的交换，使得 i 左边的数据均小于基准元素，i 和 j 之间的数据均大于基准元素，最终成功将数组以基准元素为界分为两个子数组。

针对如何选择基准元素，有四种不同的选择方式：

1.总是使用第一个元素作为基准元素

2.总是使用最后一个元素作为基准元素

3.使用一个随机元素作为基准元素

4.使用“三取中”的方法选取基准元素

分别实现这四种选择方式，并通过样例输入进行比较。

2. 统计比较次数

为了统计总比较次数，设置全局变量 compare_num，在每次调用 QuickSort 函数时，该变量增加当前数组长度减 1 即可（因为基准元素会和除它本身之外的每个元素发生比较）

二、程序说明

通过以上的思路分析，设计可递归调用的 QuickSort 函数，以及重要的 Partition 函数。并针对四种不同的基准元素选择方式，设计 ChoosePivot1、

ChoosePivot2、ChoosePivot3 和 ChoosePivot4 函数对应四种方式，可以在 QuickSort 函数中进行改变。

另外设置 N 表示待输入数组的长度，可以直接在 define 语句中进行修改。

程序详见“QuickSort.cpp”。

三、测试结果及分析

在 http://www.algorithmsilluminated.org/网页上下载样例输入，进行测试

1. 总是使用第一个元素作为基准元素

测试样例 1

```
504 609 2148 3153 5469 6324 7017 7628 7742 9058
The total number of comparisions is 25
PS D:\编程>
```

测试样例 2

```
123 345 392 448 479 504 540 609 872 972 1103 1190 1542 1582 1590 1605 1627 1676 1816 2014
2027 2081 2148 2327 2480 2655 2667 2770 2795 2951 3062 3130 3138 3153 3171 3404 3490 3513
3517 3816 3859 3915 4007 4092 4218 4264 4307 4388 4596 4715 4778 4869 4984 5002 5254 5469
5498 5565 5689 5697 5739 5780 5859 6108 6183 6237 6324 6551 6558 6756 6788 6798 6837 7017
7173 7432 7538 7578 7628 7671 7742 7923 8121 8143 8439 8666 8737 8887 8910 9058 9079 9172
9293 9310 9341 9572 9593 9595 9771 9832
The total number of comparisions is 620
PS D:\编程>
```

挑战数据集:

```
64 9765 9766 9767 9768 9769 9770 9771 9772 9773 9774 9775 9776 9777 9778 9779 9780 9781 9782 9783 97
84 9785 9786 9787 9788 9789 9790 9791 9792 9793 9794 9795 9796 9797 9798 9799 9800 9801 9802 9803 98
04 9805 9806 9807 9808 9809 9810 9811 9812 9813 9814 9815 9816 9817 9818 9819 9820 9821 9822 9823 98
24 9825 9826 9827 9828 9829 9830 9831 9832 9833 9834 9835 9836 9837 9838 9839 9840 9841 9842 9843 98
44 9845 9846 9847 9848 9849 9850 9851 9852 9853 9854 9855 9856 9857 9858 9859 9860 9861 9862 9863 98
64 9865 9866 9867 9868 9869 9870 9871 9872 9873 9874 9875 9876 9877 9878 9879 9880 9881 9882 9883 98
84 9885 9886 9887 9888 9889 9890 9891 9892 9893 9894 9895 9896 9897 9898 9899 9900 9901 9902 9903 99
04 9905 9906 9907 9908 9909 9910 9911 9912 9913 9914 9915 9916 9917 9918 9919 9920 9921 9922 9923 99
24 9925 9926 9927 9928 9929 9930 9931 9932 9933 9934 9935 9936 9937 9938 9939 9940 9941 9942 9943 99
44 9945 9946 9947 9948 9949 9950 9951 9952 9953 9954 9955 9956 9957 9958 9959 9960 9961 9962 9963 99
64 9965 9966 9967 9968 9969 9970 9971 9972 9973 9974 9975 9976 9977 9978 9979 9980 9981 9982 9983 99
84 9985 9986 9987 9988 9989 9990 9991 9992 9993 9994 9995 9996 9997 9998 9999 10000
The total number of comparisions is 162085
PS D:\编程>
```

2. 总是使用最后一个元素作为基准元素

测试样例 1

```
504 609 2148 3153 5469 6324 7017 7628 7742 9058
The total number of comparisions is 31
```

测试样例 2

```
123 345 392 448 479 504 540 609 872 972 1103 1190 1542 1582 1590 1605 1627 1676 1816 2014 2027 2081
2148 2327 2480 2655 2667 2770 2795 2951 3062 3130 3138 3153 3171 3404 3490 3513 3517 3816 3859 3915
4007 4092 4218 4264 4307 4388 4596 4715 4778 4869 4984 5002 5254 5469 5498 5565 5689 5697 5739 5780
5859 6108 6183 6237 6324 6551 6558 6756 6788 6798 6837 7017 7173 7432 7538 7578 7628 7671 7742 7923
8121 8143 8439 8666 8737 8887 8910 9058 9079 9172 9293 9310 9341 9572 9593 9595 9771 9832
The total number of comparisions is 573
PS D:\编程>
```

挑战数据集

```
42 9843 9844 9845 9846 9847 9848 9849 9850 9851 9852 9853 9854 9855 9856 9857 9858 9859 9860 9861 98
62 9863 9864 9865 9866 9867 9868 9869 9870 9871 9872 9873 9874 9875 9876 9877 9878 9879 9880 9881 98
82 9883 9884 9885 9886 9887 9888 9889 9890 9891 9892 9893 9894 9895 9896 9897 9898 9899 9900 9901 99
02 9903 9904 9905 9906 9907 9908 9909 9910 9911 9912 9913 9914 9915 9916 9917 9918 9919 9920 9921 99
22 9923 9924 9925 9926 9927 9928 9929 9930 9931 9932 9933 9934 9935 9936 9937 9938 9939 9940 9941 99
42 9943 9944 9945 9946 9947 9948 9949 9950 9951 9952 9953 9954 9955 9956 9957 9958 9959 9960 9961 99
62 9963 9964 9965 9966 9967 9968 9969 9970 9971 9972 9973 9974 9975 9976 9977 9978 9979 9980 9981 99
82 9983 9984 9985 9986 9987 9988 9989 9990 9991 9992 9993 9994 9995 9996 9997 9998 9999 10000
The total number of comparisions is 164123
PS D:\编程>
```

3. 使用一个随机元素作为基准元素

测试样例 1

```
504 609 2148 3153 5469 6324 7017 7628 7742 9058
The total number of comparisions is 19
```

测试样例 2

```
123 345 392 448 479 504 540 609 872 972 1103 1190 1542 1582 1590 1605 1627 1676 1816 2014 2027 2081
2148 2327 2480 2655 2667 2770 2795 2951 3062 3130 3138 3153 3171 3404 3490 3513 3517 3816 3859 3915
4007 4092 4218 4264 4307 4388 4596 4715 4778 4869 4984 5002 5254 5469 5498 5565 5689 5697 5739 5780
5859 6108 6183 6237 6324 6551 6558 6756 6788 6798 6837 7017 7173 7432 7538 7578 7628 7671 7742 7923
8121 8143 8439 8666 8737 8887 8910 9058 9079 9172 9293 9310 9341 9572 9593 9595 9771 9832
The total number of comparisions is 627
```

挑战数据集

```
62 9863 9864 9865 9866 9867 9868 9869 9870 9871 9872 9873 9874 9875 9876 9877 9878 9879 9880 9881 98
82 9883 9884 9885 9886 9887 9888 9889 9890 9891 9892 9893 9894 9895 9896 9897 9898 9899 9900 9901 99
02 9903 9904 9905 9906 9907 9908 9909 9910 9911 9912 9913 9914 9915 9916 9917 9918 9919 9920 9921 99
22 9923 9924 9925 9926 9927 9928 9929 9930 9931 9932 9933 9934 9935 9936 9937 9938 9939 9940 9941 99
42 9943 9944 9945 9946 9947 9948 9949 9950 9951 9952 9953 9954 9955 9956 9957 9958 9959 9960 9961 99
62 9963 9964 9965 9966 9967 9968 9969 9970 9971 9972 9973 9974 9975 9976 9977 9978 9979 9980 9981 99
82 9983 9984 9985 9986 9987 9988 9989 9990 9991 9992 9993 9994 9995 9996 9997 9998 9999 10000
The total number of comparisions is 156419
```

4. 使用“三取中”的方法选取基准元素

测试样例 1

如果不计算选择中位元素作为 pivot 时的比较次数:

```
504 609 2148 3153 5469 6324 7017 7628 7742 9058
The total number of comparisions is 21
```

如果计算：

```
504 609 2148 3153 5469 6324 7017 7628 7742 9058
The total number of comparisions is 35
```

测试样例 2

如果不计算选择中位元素作为 pivot 时的比较次数：

```
123 345 392 448 479 504 540 609 872 972 1103 1190 1542 1582 1590 1605 1627 1676 1816 2014 2027 2081
2148 2327 2480 2655 2667 2770 2795 2951 3062 3130 3138 3153 3171 3404 3490 3513 3517 3816 3859 3915
4007 4092 4218 4264 4307 4388 4596 4715 4778 4869 4984 5002 5254 5469 5498 5565 5689 5697 5739 5780
5859 6108 6183 6237 6324 6551 6558 6756 6788 6798 6837 7017 7173 7432 7538 7578 7628 7671 7742 7923
8121 8143 8439 8666 8737 8887 8910 9058 9079 9172 9293 9310 9341 9572 9593 9595 9771 9832
The total number of comparisions is 502
```

如果计算：

```
123 345 392 448 479 504 540 609 872 972 1103 1190 1542 1582 1590 1605 1627 1676 1816 2014 2027 2081
2148 2327 2480 2655 2667 2770 2795 2951 3062 3130 3138 3153 3171 3404 3490 3513 3517 3816 3859 3915
4007 4092 4218 4264 4307 4388 4596 4715 4778 4869 4984 5002 5254 5469 5498 5565 5689 5697 5739 5780
5859 6108 6183 6237 6324 6551 6558 6756 6788 6798 6837 7017 7173 7432 7538 7578 7628 7671 7742 7923
8121 8143 8439 8666 8737 8887 8910 9058 9079 9172 9293 9310 9341 9572 9593 9595 9771 9832
The total number of comparisions is 642
PS D:\编程>
```

挑战数据集：

如果不计算选择中位元素作为 pivot 时的比较次数：

```
62 9863 9864 9865 9866 9867 9868 9869 9870 9871 9872 9873 9874 9875 9876 9877 9878 9879 9880 9881 98
82 9883 9884 9885 9886 9887 9888 9889 9890 9891 9892 9893 9894 9895 9896 9897 9898 9899 9900 9901 99
02 9903 9904 9905 9906 9907 9908 9909 9910 9911 9912 9913 9914 9915 9916 9917 9918 9919 9920 9921 99
22 9923 9924 9925 9926 9927 9928 9929 9930 9931 9932 9933 9934 9935 9936 9937 9938 9939 9940 9941 99
42 9943 9944 9945 9946 9947 9948 9949 9950 9951 9952 9953 9954 9955 9956 9957 9958 9959 9960 9961 99
62 9963 9964 9965 9966 9967 9968 9969 9970 9971 9972 9973 9974 9975 9976 9977 9978 9979 9980 9981 99
82 9983 9984 9985 9986 9987 9988 9989 9990 9991 9992 9993 9994 9995 9996 9997 9998 9999 10000
The total number of comparisions is 138382
```

如果计算：

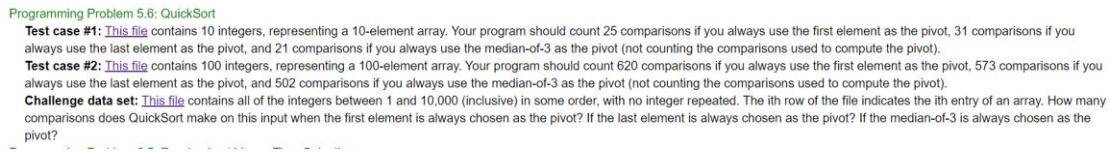
```
62 9863 9864 9865 9866 9867 9868 9869 9870 9871 9872 9873 9874 9875 9876 9877 9878 9879 9880 9881 98
82 9883 9884 9885 9886 9887 9888 9889 9890 9891 9892 9893 9894 9895 9896 9897 9898 9899 9900 9901 99
02 9903 9904 9905 9906 9907 9908 9909 9910 9911 9912 9913 9914 9915 9916 9917 9918 9919 9920 9921 99
22 9923 9924 9925 9926 9927 9928 9929 9930 9931 9932 9933 9934 9935 9936 9937 9938 9939 9940 9941 99
42 9943 9944 9945 9946 9947 9948 9949 9950 9951 9952 9953 9954 9955 9956 9957 9958 9959 9960 9961 99
62 9963 9964 9965 9966 9967 9968 9969 9970 9971 9972 9973 9974 9975 9976 9977 9978 9979 9980 9981 99
82 9983 9984 9985 9986 9987 9988 9989 9990 9991 9992 9993 9994 9995 9996 9997 9998 9999 10000
The total number of comparisions is 153452
```

四、算法比较和分析

将上述运行结果制成如下表：

选择方法		输入		
		样例 1	样例 2	挑战数据集
第一个元素		25	620	162085
最后一个元素		31	573	164123
随机		19	627	156419
三取中	不计算取中时的比较次数	21	502	138382
	计算取中时的比较次数	35	642	153452

可以看到其结果与官网所给的数据是一致的，证明程序正确无误。


Programming Problem 5.6: QuickSort

Test case #1: This file contains 10 integers, representing a 10-element array. Your program should count 25 comparisons if you always use the first element as the pivot, 31 comparisons if you always use the last element as the pivot, and 21 comparisons if you always use the median-of-3 as the pivot (not counting the comparisons used to compute the pivot).

Test case #2: This file contains 100 integers, representing a 100-element array. Your program should count 620 comparisons if you always use the first element as the pivot, 573 comparisons if you always use the last element as the pivot, and 502 comparisons if you always use the median-of-3 as the pivot (not counting the comparisons used to compute the pivot).

Challenge data set: This file contains all of the integers between 1 and 10,000 (inclusive) in some order, with no integer repeated. The ith row of the file indicates the ith entry of an array. How many comparisons does QuickSort make on this input when the first element is always chosen as the pivot? If the last element is always chosen as the pivot? If the median-of-3 is always chosen as the pivot?

从以上结果中可以看出，就整体效果而言，随机选取法和三取中法都比固定位置（第一个或最后一个）选取法要好。因为固定位置选取法在最坏情况（数组已经有序）时，每次划分只能使待排序序列减一，快速排序沦为起泡排序，时间复杂度为Θ(n^2)。而随机选取法中，由于枢轴的位置是随机的，那么产生的分割也不会总是会出现劣质的分割，所以随机法可以对于绝大多数输入数据达到O(nlogn）的期望时间复杂度。三取中法则是通过首尾中间三个数的中间值代替整个数组的中间值，虽然不能完全与中间值相等，但是仍然能比较有效的分割。不过“取中”操作要进行很多次，相应的也会增加很多比较次数（尽管可以不计算）。

综上，QuickSort 算法实现时还是选择随机选取 pivot 或“三取中”法选取 pivot 的效果更好。