

学号	2022212080	算法思路 (30%)	编码实现与 算法分析 (50%)	实验报告 (20%)	得分
姓名	刘纪彤				
评语					

# 《算法设计与分析》实验报告

## 实验 2 分治算法实验

### 一、实验目的

1. 加深对分治法算法设计的理解，包括其基本原理和递归调用的机制。
2. 学习如何将大问题分解为小问题，并独立解决这些小问题。
3. 掌握分治法算法的三个步骤：分解、解决、合并。
4. 分析分治法算法的时间复杂度和空间复杂度。
5. 提高编程能力和复杂问题解决能力。

### 二、实验内容(题目)

用分治法实现一组无序序列的两路合并排序和快速排序。要求清楚合并排序及快速排序的基本原理，编程实现分别用这两种方法将输入（或者利用随机函数生成）的一组（不得少于 50 个数据）无序序列排序为有序序列后输出。

### 三、算法设计思路

二路合并排序算法思路，利用递归，对左右两个分别进行递归调用进行排序，同时再把已经大体上算好的进行合并。

快速排序就是，利用分治的策略，将比 Key 小的放在左侧，key 大的放在右侧，按照递归分治的思想最终输出一个排好序的值

### 四、各功能模块设计

当他

```
#include <bits/stdc++.h>
using namespace std;

void merge(vector<int> &b,int l,int mid,int r)
{
    vector<int> temp(r-l+1);
    int i=l;
    int j=mid+1;
    int k=0;
```

```

    while((i<=mid)&&(j<=r))
    {
        if(b[i]<=b[j]) temp[k++]=b[i++];
        else temp[k++]=b[j++];
    }
    while(i<=mid) temp[k++]=b[i++];
    while(j<=r) temp[k++]=b[j++];
    for(i=l,k=0;i<=r;i++,k++) b[i]=temp[k];//将临时存储还给
}

void mergeSort(vector<int> &b,int l,int r)
{
    if(l<r){
        int mid=l+(r-l)/2;
        mergeSort(b,l,mid);
        mergeSort(b,mid+1,r);
        merge(b,l,mid,r);
    }
}

void quickSort(vector<int> &b,int l,int r)
{
    if(l>=r) return;
    int i=l;
    int j=r;
    int key=b[l];//关键值
    while(i<j)
    {
        while(i<j&& b[j]>=key) j--;
        b[i]=b[j];
        while(i<j&& b[i]<=key) i++;
        b[j]=b[i];
    }
    b[i]=key;
    quickSort(b,l,i-1);
    quickSort(b,i+1,r);
}

int main(){
    vector<int> a;
    int n;
    cin>>n;
    for(int i=0;i<n;i++)
    {
        a.push_back(((float)(1.0*rand()/(RAND_MAX + 1)))*(50));
    }
    cout<<"排序前数组: ";

```

```

for(int b:a)
{
    cout<<b<<" ";
}
cout<<endl;
int ch;
cout<<"请选择排序方式：1.归并排序 2.快速排序"<<endl;
cin>>ch;
if(ch==1) mergeSort(a,0,n-1);
if(ch==2) quickSort(a,0,n-1);
cout<<"排序后数组：";
for(int b:a)
{
    cout<<b<<" ";
}
cout<<endl;
}

```

## 五、运行结果与分析

```

PS F:\we will study> cd 'f:\we will study'
PS F:\we will study> & .\1.exe
102
排序前数组： 2686976 1210253312 415105024 1736704000 1256259584 1030488064 752222208 1924005888 1766981632 1063272704 373882880 1846510720 1525743616 1102774272 652804096 32178176 196280320 782630912 316342272 35625
3696 2122776576 957088744 255721472 10827008 19136512 811466752 1141702656 1226571776 1292238848 1303838720 356974592 1423835136 968032256 756154368 122486784 1304952832 1682112512 1723531264 1116405760 648413184 1
881079808 1560477696 2052718592 1987993488 1158217728 305659904 992280576 585348096 1851588608 450101248 1674248192 1811677184 2140536832 2146762752 1313164832 842727424 571670528 638386176 1804140544 509870808 8071
41376 198981760 1454243840 128717312 18874368 1973820816 592465440 586022912 1262485504 1484259328 1798701056 1560884480 1941367040 440991744 1597112320 1085977008 983433216 2038235136 1598619648 232521728 12864061
44 827260928 1578369024 1307705344 1229193216 775946240 325451776 483393536 912962016 1724121088 1110641984 2125922304 1613889536 742064128 362872832 1411514368 1856309248 136445952 1502674944 1084030976 316735488
2039152640
请选择排序方式： 1.归并排序 2.快速排序
1
排序后数组： 2686976 10027008 18874368 19136512 32178176 509870808 120717312 122486784 136445952 196280320 198981760 232521728 255721472 305659904 316342272 316735488 325451776 356253696 356974592 362872832 373882880
415105024 440991744 450101248 483393536 505348096 571670528 586022912 592465440 638386176 648413184 652804096 742064128 752222208 756154368 775946240 782630912 807141376 811466752 827260928 842727424 912982016 957
087744 968032256 983433216 992280576 1005977008 1030488064 1041367040 1056309248 1084030976 1102774272 1110641984 1116405760 1141702656 1158217728 1210253312 1226571776 1229193216 1256259584 1262485504 1286406144 1
292238848 1303838720 1304952832 1307705344 1313164832 1411514368 1423835136 1454243840 1484259328 1502674944 1525743616 1560884480 1568477696 1578369024 1597112320 1598619648 1603272704 1613889536 1674248192 168211
2512 1723531264 1724121088 1736704000 1766981632 1798701056 1804140544 1811677184 1846510720 1851588608 1881079808 1924005888 1973020816 198703488 2038235136 2039152640 2052718592 2122776576 2125922304 2140536832
2146762752
请选择排序方式： 1.归并排序 2.快速排序
2
排序前数组： 2686976 1210253312 415105024 1736704000 1256259584 1030488064 752222208 1924005888 1766981632 1063272704 373882880 1846510720 1525743616 1102774272 652804096 32178176 196280320 782630912 316342272 35625
3696 2122776576 957088744 255721472 10827008 19136512 811466752 1141702656 1226571776 1292238848 1303838720 356974592 1423835136 968032256 756154368 122486784 1304952832 1682112512 1723531264 1116405760 648413184 1
881079808 1560477696 2052718592 1987993488 1158217728 305659904 992280576 585348096 1851588608 450101248 1674248192 1811677184 2140536832 2146762752 1313164832 842727424 571670528 638386176 1804140544 509870808 8071
41376 198981760 1454243840 128717312 18874368 1973820816 592465440 586022912 1262485504 1484259328 1798701056 1560884480 1941367040 440991744 1597112320 1085977008 983433216 2038235136 1598619648 232521728 12864061
44 827260928 1578369024 1307705344 1229193216 775946240 325451776 483393536 912962016 1724121088 1110641984 2125922304 1613889536 742064128 362872832 1411514368 1856309248 136445952 1502674944 1084030976 316735488
2039152640
请选择排序方式： 1.归并排序 2.快速排序
2
排序后数组： 2686976 10027008 18874368 19136512 32178176 509870808 120717312 122486784 136445952 196280320 198981760 232521728 255721472 305659904 316342272 316735488 325451776 356253696 356974592 362872832 373882880
415105024 440991744 450101248 483393536 505348096 571670528 586022912 592465440 638386176 648413184 652804096 742064128 752222208 756154368 775946240 782630912 807141376 811466752 827260928 842727424 912982016 957
087744 968032256 983433216 992280576 1005977008 1030488064 1041367040 1056309248 1084030976 1102774272 1110641984 1116405760 1141702656 1158217728 1210253312 1226571776 1229193216 1256259584 1262485504 1286406144 1
292238848 1303838720 1304952832 1307705344 1313164832 1411514368 1423835136 1454243840 1484259328 1502674944 1525743616 1560884480 1568477696 1578369024 1597112320 1598619648 1603272704 1613889536 1674248192 168211
2512 1723531264 1724121088 1736704000 1766981632 1798701056 1804140544 1811677184 1846510720 1851588608 1881079808 1924005888 1973020816 198703488 2038235136 2039152640 2052718592 2122776576 2125922304 2140536832
2146762752

```

2-1

如图所示 2-1 对于排好序的输出结果均为单调递增，故输出都是正确的值。

## 六、实验总结

通过本次实验我已经了解并掌握了分治的基本逻辑，结合前述所学的递归知识，能够将分治的思想利用已学的递归结合在一起，能够使用分治的思路解决实际应用中的问题，受益匪浅。