# 实验一：递归与分治

**一、实验目的**

理解递归算法的思想和递归程序的执行过程，并能熟练编写递归程序。

掌握分治算法的思想，对给定的问题能设计出分治算法予以解决。

**二、实验内容**

段合并排序算法：

将数组划分为个子数组，每个子数组有个元素。然后递归地对分割后的子数组进行排序，最后将所得到的个排好序的子数组合并排序。

1. **分治算法基本思想**

**将一个大的任务分成若干个小任务分别完成，再合并起来解决**

1. **实验过程**

**输入n,输入a数组的各个元素**

**通过sort函数进行排序，首先分成**个子数组，通过递归对这个子数组分别调用sort函数，递归结束的条件是数组内元素<=1。接下来合并个排好序的子数组，暂存到tmp，最后赋值回a数组。

/\*

 \* @Author: Kaizyn

 \* @Date: 2021-05-10 08:49:16

 \* @LastEditTime: 2021-05-10 09:06:17

 \*/

#include <bits/stdc++.h>

using namespace std;

int n;

vector<int> a;

void sort(int l, int r) {

  if (r-l <= 1) return;

  int block = sqrt(r-l)+.5, num = (r-l+block-1)/block;

  vector<int> head(num), tail(num), tmp(r-l);

  for (int i = 0; i < num; ++i) {

    head[i] = l+i\*block;

    tail[i] = min(head[i]+block, r);

    sort(head[i], tail[i]);

  }

  for (int i = 0, k; i < r-l; ++i) {

    k = -1;

    for (int j = 0; j < num; ++j) {

      if (head[j] >= tail[j]) continue;

      if (k == -1 || a[head[j]] < a[head[k]]) k = j;

    }

    tmp[i] = a[head[k]++];

  }

  for (int i = 0; i < r-l; ++i) a[l+i] = tmp[i];

  // memcpy(a.begin()+l, tmp.begin(), sizeof(int)\*(r-l));

}

signed main() {

#ifdef ONLINE\_JUDGE

  ios::sync\_with\_stdio(false); cin.tie(nullptr); cout.tie(nullptr);

#endif

  cin >> n;

  a.resize(n);

  for (int i = 0; i < n; ++i) cin >> a[i];

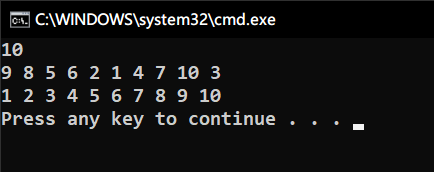
  sort(0, n);

  for (int i = 0; i < n; ++i) cout << a[i] << " \n"[i==n-1];

  return 0;

}

1. **实验结果**



合并个排好序的子数组时，依次确定n个位置元素的值，每次都需要比较个排好序的子数组的第一个元素。一遍的复杂度时O(n),因为是开根，所以递归层数很少，n=1,000,000时也只有5层左右。又如第二层单个函数复杂度O(√n\*√√n)共有√n次，所以第二层总体复杂度O(n\*√√n)是远小于第一层的。因此该排序算法的时间复杂度为 O(n)