# 实验三：贪心算法

**一、实验目的**

（1）掌握贪心算法的基本思想；

（2）能使用贪心算法求解一些相关问题；

**二、实验内容**

1、给定k个排好序的序列s1,s2,...,sk，用2路合并算法将这k个序列合并成一个序列。假设所采用的2路合并算法合并2个长度为m和n的序列需要m+n-1次比较。设计一个算法确定2路合并次序，使所需的总比较次数最少。

1. **贪心算法基本思想**

贪心算法是指在对问题进行求解时，总是做出在当前看来的是最好的选择，不从整体最优上加以考虑。也就是说每次总是得到当前问题的局部最优解，最后从局部的最优解来构造总问题的最优解。其中如何定义和得到问题的局部最优解，依赖于贪心策略的选择。

如何定义本问题的贪心策略选择，根据问题的描述，2路合并算法是指，在k个待合并的有序序列中，每次选择两个序列进行有序合并，并将这两个序列从原来的序列组中剔除。然后再将新序列加入到原先的序列组中。依次重复以上操作，直到最后得到一个完整的有序序列。

类比数据结构中赫夫曼树的构造，也是一种贪心策略的选择。而这次实验题目贪心算法的设计和赫曼树的构造有异曲同工之妙。因此可以这样定义本次实验题目的贪心策略：

为2路合并算法添加规则：每次在序列组中选择长度最小的两个有序序列进行合并，得到当前问题的最少比较次数。重复以上选择操作，最后可以得到总问题的最优解。

1. **实验过程**

**实验所用开发平台为vscode**

**实验代码描述语言为c++**

#include <stdio.h>

#include <algorithm>

using namespace std;

//贪心算法计算最优值

//贪心策略：每次选长度最小的两个序列合并,得到最少比较次数。

int greedyAlgorithm(int\* a, int size)

{

int\* b;

int minSum = 0;

b = (int\*)malloc(sizeof(int) \* size);

for (int i = 0; i < size; i++)

{

b[i] = a[i];

}

while (size > 1)

{

sort(b, b + size); //对数组地址从 b 到 b+m-1 的元素进行从小到大排序

printf("{%d,%d} ",b[0],b[1]);

b[0] = b[0] + b[1]; //选择所有序列中长度最小的两个序列，合并成一个新的序列

minSum += b[0] - 1; //计算比较次数

for (int i = 1; i < size - 1; i++)

{

b[i] = b[i + 1];

}

size--;

}

printf("\n");

return minSum;

}

int main()

{

int n;

int\* a;

int minSum;

printf("Please enter the number of ordered sequences to be merged:\n");

scanf("%d",&n);

a = (int\*)malloc(sizeof(int) \* n);

printf("Please enter the length of each ordered sequence in turn:\n");

for (int i = 0; i < n; i++)

{

scanf("%d",&a[i]);

}

printf("The merge order of an ordered sequence is:\n");

minSum = greedyAlgorithm(a, n);

printf("The minimum total number of comparisons determined by the greedy algorithm is: \n");

printf("%d\n",minSum);

system("pause");

return 0;

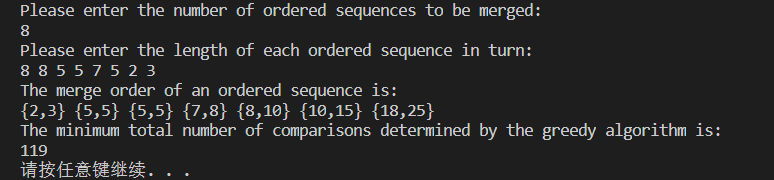
}

1. **实验结果**

测试案例1：

有序序列个数：8

每个有序序列长度分别为： 8、8、5、5、7、5、2、3



测试案例2：

有序序列个数：10

每个有序序列长度分别为： 1、2、3、4、5、6、7、8、9、10

