**算法实习报告：实验一 段合并排序算法的实现**

题目： 段合并排序算法的实现

1. **需求分析**

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. **概要设计**

实现上述策略的合并排序算法如下：

template<class T>

void mergesort(T \*a, int left, int right) {

if (left < right) {

int j = (int)*sqrt*(right - right + 1);

if (j > 1) {

for (int i = 0; i < j; i++) {

mergesort(a, left + i\*j, left + (i + 1)\*j - 1);

}

mergesort(a, left + j\*j; right);

}

mergeall(a, left, right);

}

}

其中，算法mergeall合并根号n个排好序的数组段。

1. **详细设计**

#include "iostream"

using namespace *std*;

#define MUX 30

void mergesort(int a[], int left, int right);

void mergeall(int a[], int left, int right);

void merge(int a[], int b[], int left, int mid, int right);

void copy(int a[], int b[], int left, int right);

void createdata(int a[]);

void mergesort(int a[], int left, int right) {

if (left < right) {

int j = (int)*sqrt*(right - left + 1);

if (j > 1) {

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

mergesort(a, left + i \*j, left + (i + 1)\*j - 1);

mergesort(a, left + j\*j, right);

}

mergeall(a, left, right);

}

}

void mergeall(int a[], int left, int right) {

int \*b = new int[right - left + 1];

for (int k = 0; k < right - left + 1; k++) {

b[k] = 0;

}

int j = (int)*sqrt*(right - left + 1);

for (int i = 0; i < j; i++) {

merge(a, b, left, left + i\*j, left + (i + 1)\*j - 1);

//将a数组合并到数组b

copy(a, b, left, left + (i + 1)\*j - 1); //复制回数组a

}

merge(a, b, left, left + j\*j, right);

copy(a, b, left, right);

delete b;

}

void merge(int a[], int b[], int left, int mid, int right) {

int i = left, j = mid, k = 0;

while ((i < mid) && (j <= right)) {

if (a[i] <= a[j]) {

b[k] = a[i];

i++;

}

else {

b[k] = a[j];

j++;

}

k++;

}

while (i < mid) {

b[k] = a[i];

k++;

i++;

}

while (j <= right) {

b[k] = a[j];

k++;

j++;

}

}

void copy(int a[], int b[], int left, int right) {

for (int i = left; i <= right; i++) {

a[i] = b[i - left];

}

}

void main() {

int a[MUX];

createdata(a);

mergesort(a, 0, MUX - 1);

*cout* << "排序后的数组如下：" << *endl*;

for (int i = 0; i < MUX; i++) {

*cout* << a[i] << ' ';

}

*cout* << *endl*;

}

void createdata(int a[]) {

*cout* << "排序前的随机数组如下：" << *endl*;

for (int i = 0; i < MUX; i++) {

a[i] = *rand*() % 100 + 1; //产生1~100的随机数

*cout* << a[i] << ' ';

}

*cout* << *endl*;

}

1. **调试分析**

本程序在最坏的条件下，算法mergeall需要O(nlogn)时间。因此递归式的解为T(n)=O(nlogn)。

1. **用户手册**
2. 用户可以通过修改 MUX 常量，来更改需要进行合并排序的数组大小。
3. 系统可以自动对 MUX 长度的随机数组，使用合并排序算法进行从小到大排序，并输出。
4. **测试结果**

我们设置 MUX 值为默认值30。

终端输出结果如下：

