

**实验（实习）报告**

实验(实习)名称：实验1

日期：2022.5.3

学院：应用技术学院

专业：计算机科学与技术

班级：1班

姓名：成凯

学号：201833050025

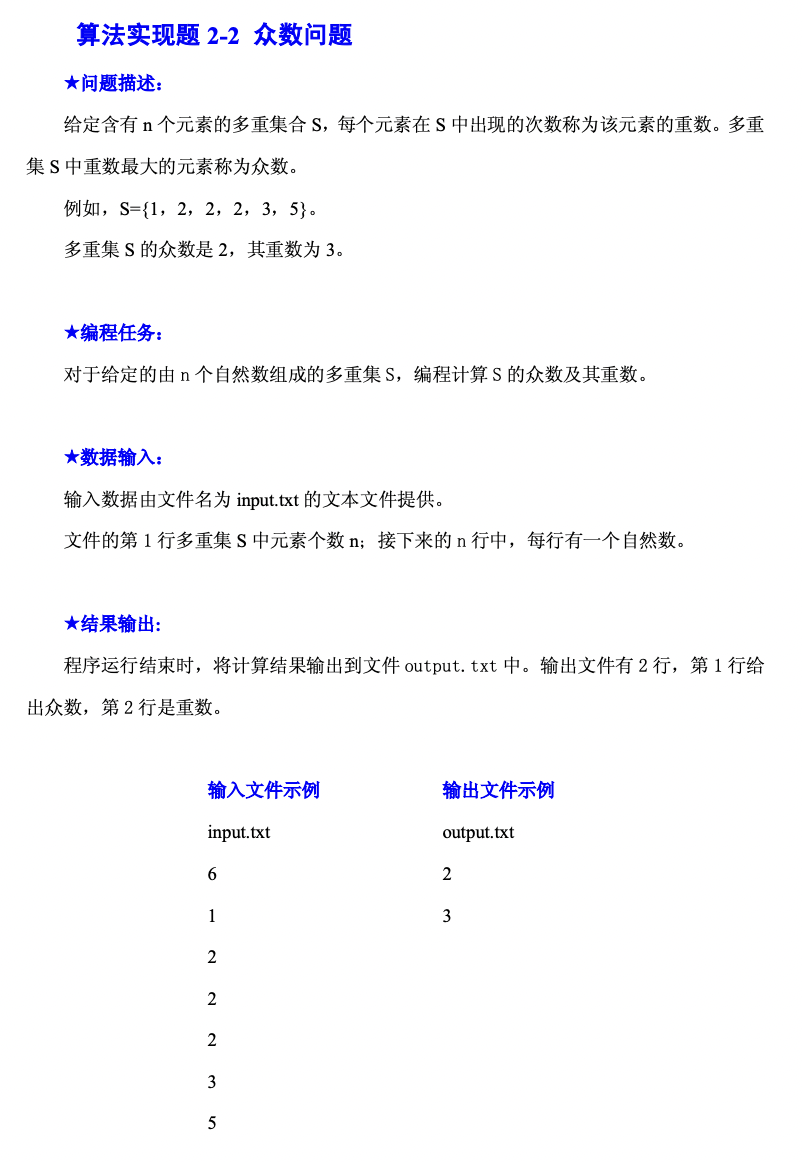
指导教师：庞亚伟

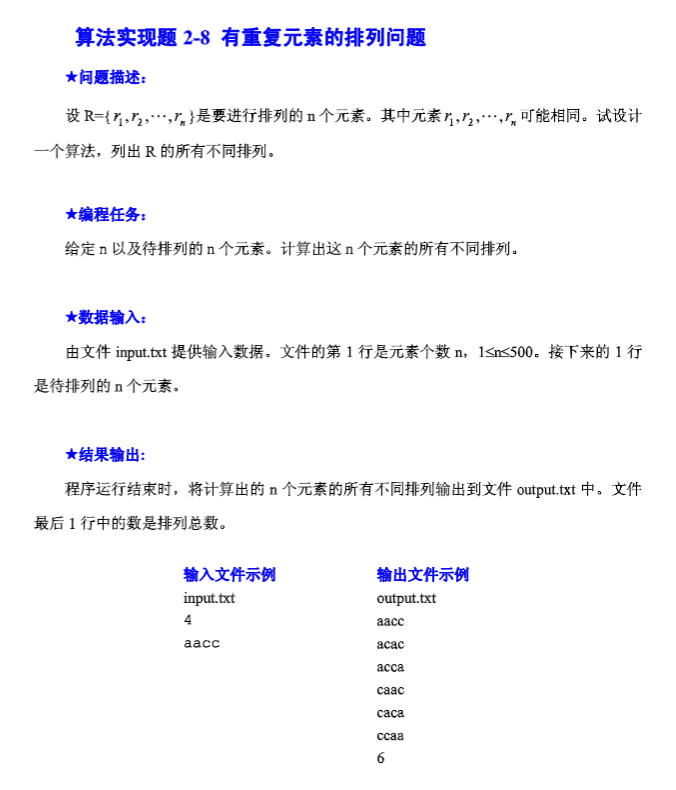
1. 实验目的

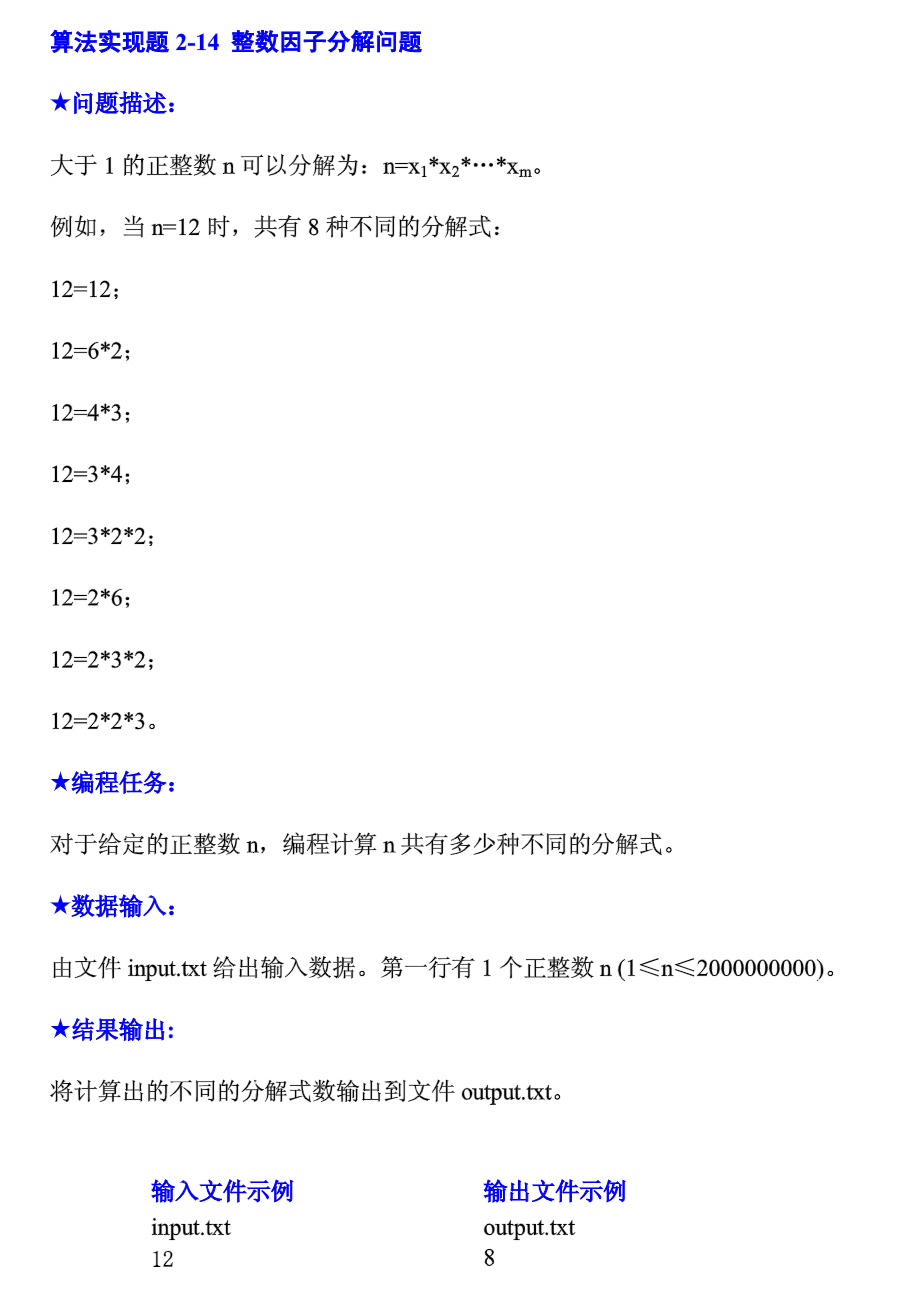
（1）理解掌握所学的第二章的总体算法思想；

（2）学会使用递归与分置解决部分算法问题；

1. 实验内容







1. 实验步骤和实验结果

算法一实现代码：

#include<stdio.h>

#include<string.h>

void sort(int buf[], int len) {

int i, j, k;

for (i = 0; i < len - 1; i++) {

k = i;

for (j = i + 1; j < len; j++) {

if (buf[k] > buf[j]) {

k = j;

}

}

if (k != i) {

buf[k] = buf[k] ^ buf[i];

buf[i] = buf[k] ^ buf[i];

buf[k] = buf[k] ^ buf[i];

}

}

}

int main() {

int i, j, k, max, m, buf[100], buf1[100], buf2[100];

bzero(buf, sizeof(buf));

bzero(buf1, sizeof(buf1));

bzero(buf2, sizeof(buf2));

scanf("%d", &m);

for (i = 0; i < m; i++) {

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

}

sort(buf, m);

j = 0;

for (i = 0; i < m; i++) {

k = i;

buf2[j] = buf[i];

while (buf[k] == buf[i]) {

buf1[j]++;

i++;

}

i--;

j++;

}

max = 0;

for (i = 1; i < j; i++) {

if (buf1[max] < buf1[i]) {

max = i;

}

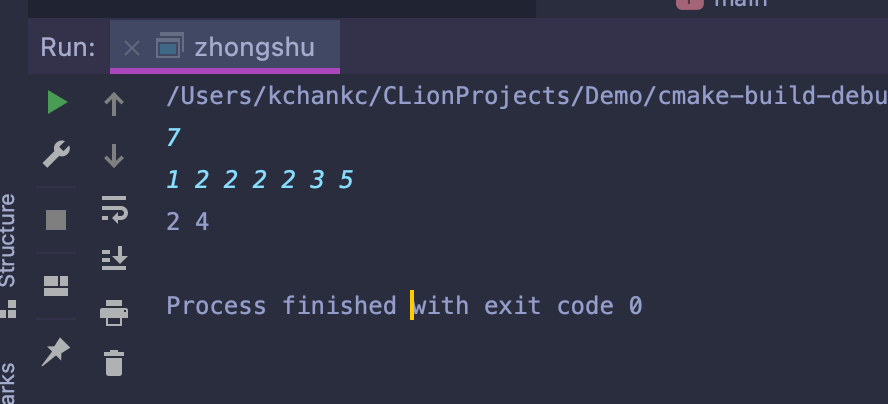
}

printf("%d %d\n", buf2[max], buf1[max]);

return 0;

}

算法一运行结果：



算法二实现代码：

#include<iostream>

using namespace std;

char xx[6666];

int cnt = 0;

int findsame(int k, int m) {

for (int i = k; i < m; i++) {

if (xx[i] == xx[m])

return 1;

}

return 0;

}

void perm(int k, int m) {

if (k == m) {

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

cout << xx[i];

}

cout << endl;

cnt++;

return;

}

for (int i = k; i <= m; i++) {

if (findsame(k, i))

continue;

swap(xx[k], xx[i]);

perm(k + 1, m);

swap(xx[k], xx[i]);

}

}

int main() {

int n;

cout << "请输入元素个数：";

cin >> n;

cout << "请输入" << n << "个不同的元素：";

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

cin >> xx[i];

perm(0, n - 1);

cout << "count=" << cnt;

return 0;

}

算法二运行结果：



算法三实现代码：

#include<stdio.h>

#include<stdlib.h>

#define maxsize 100

int count = 1;

void f(int\* a, int m) {

int i, j, k;

int\* p;

for (i = 0; i <= m; i++) {

k = -1;

p = (int\*) malloc(sizeof(int) \* maxsize);

for (j = 2; j < a[i]; j++) {

if (a[i] % j == 0) {

k++;

p[k] = j;

}

}

if (k > -1) {

f(p, k);

count = count + k + 1;

}

free(p);

}

return;

}

int main() {

int\* a;

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

scanf("%d", &a[0]);

f(a, 0);

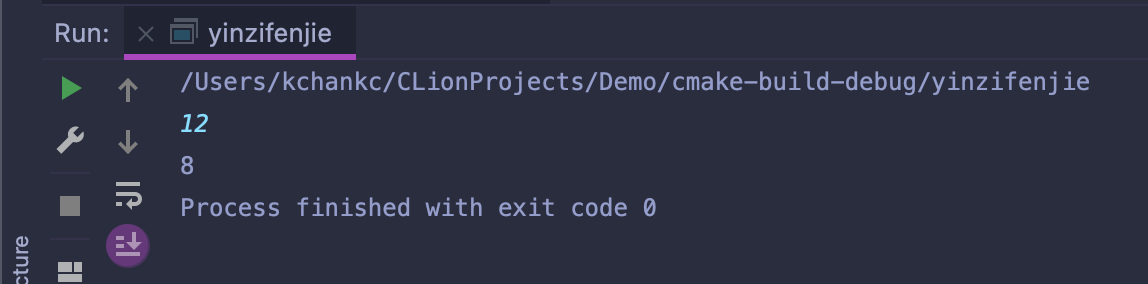
free(a);

printf("%d", count);

return 0;

}

算法三运行结果：



1. 分析与讨论

通过本次实验掌握了如何使用递归算法解决一些小问题。了解了分治算法的应用，总的来说通过本次实验受益匪浅。