数据结构实验报告

评分

满分——5分

学号：20159999 姓名：李文华 专业：计算机科学与技术

知识范畴：数组与广义表 完成日期：2017年04月21日

实验题目：基于压缩存储的半三角矩阵乘法运算的实现

实验内容及要求：

已知两个*n*阶下半三角矩阵的乘积仍为*n*阶下半三角矩阵。编程输入两个*n*阶下半三角矩阵，输出这两个矩阵的乘积。要求*n*阶下半三角矩阵采用一维数组压缩存储（即只存储下半三角）。

程序先从键盘(或字符文件)输入*n*值，建立三个矩阵的一维数组动态存储结构，然后从键盘(或字符文件)输入两个半三角矩阵，最后输出计算结果到屏幕上(或另一个字符文件中)。

例如：键盘输入为：

3

1

2 3

4 5 6

-1

-2 -3

-4 -5 -6

则输出为：

-1

-8 -9

-38 -45 -36

实验目的：掌握半三角矩阵的顺序存储结构。

数据结构设计简要描述：

用一维数组压缩存储半三角矩阵。

算法设计简要描述：

一个n\*n的下半三角矩阵中共有n\*(n+1)/2个元素，对于任意位置i,j，映射到压缩数组中对应的下标为i\*(i-1)/2+j（下标从0开始）。对应结果I,j位置的值应等于：

综合以上两点可以得到结果。

输入/输出设计简要描述：

第一行输入整数n代表为n维矩阵，接下来的2n行分别输入两个相乘的半三角矩阵。

输出结果以半三角矩阵的形式

编程语言说明：

使用Codeblocks编程。 主要代码采用C语言实现 ；动态存储分配采用C++的new和delete操作符实现；输入与输出采用C++的cin和cout流；程序注释采用C/C++规范。

主要函数说明：

int ij2num(int i, int j, int \*a)// 由坐标(i,j)得出对应压缩数组中该位置的值

int mul\_matrix(int x, int y, int \*a, int \*b)// 计算Cij的值，x，y即Cxy，a为被乘矩阵，b为乘矩阵

void print\_matrix(int \*c, int n)//打印半三角矩阵

程序测试简要报告：

***实例一：***

键盘输入为：

3

1

2 3

4 5 6

-1

-2 -3

-4 -5 -6

则输出为：

-1

-8 -9

-38 -45 -36

***结论***

程序输出结果与期望输出结果相符。

***实例二：***

键盘输入为：

4

1

2 3

4 5 6

7 8 9 10

-1

-2 -3

-4 -5 -6

-7 -8 -9 -10

则输出为：

-1

-8 -9

-38 -45 -36

-129 -149 -144 -100

***结论***

程序输出结果与期望输出结果相符。

源程序代码：

#include <iostream>

#include <fstream>

using namespace std;

/\*

由坐标(i,j)得出对应压缩数组中该位置的值

\*/

int ij2num(int i, int j, int \*a)

{

return a[i\*(i-1)/2+j];

}

/\*

计算Cij的值，x，y即Cxy，a为被乘矩阵，b为乘矩阵

\*/

int mul\_matrix(int x, int y, int \*a, int \*b)

{

int sum = 0;

for(int k = y; k <= x; k++)//计算C(x, y)的值

sum = sum + ij2num(x, k, a) \* ij2num(k, y, b);

return sum;

}

void print\_matrix(int \*c, int n)//打印半三角矩阵

{

int cnt = 0;

for(int i = 1; i <= n; i++)

{

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

cout << c[++cnt] << " ";

cout << endl;

}

}

int main()

{

ifstream cin("input.txt");

int n, length;

int cnt = 0;

int \*a, \*b, \*c;

cin >> n;

length = n \* (n + 1) / 2;

a = new int[length+1];

b = new int[length+1];

c = new int[length+1];

for(int i=1; i <= length; i++)

cin >> a[i];

for(int i=1; i <= length; i++)

cin >> b[i];

for(int i = 1; i <= n; i++)

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

{

c[++cnt] = mul\_matrix(i, j, a, b);

}

print\_matrix(a, n);

print\_matrix(b, n);

print\_matrix(c, n);

return 0;

}