目录

[1 设计题目 1](#_Toc17932)

[2 设计要求 1](#_Toc18613)

[3 总体设计 3](#_Toc10170)

[3.1 运行环境 3](#_Toc29278)

[3.2 数据来源 3](#_Toc17739)

[3.3 模块设计 3](#_Toc13859)

[3.3.1 创建稀疏矩阵 3](#_Toc24621)

[3.3.2 打印稀疏矩阵 3](#_Toc12)

[3.3.3 矩阵加法 4](#_Toc19307)

[3.3.4 矩阵减法 4](#_Toc6542)

[3.3.5 矩阵乘法 4](#_Toc19155)

[3.3.6 矩阵数乘 4](#_Toc7848)

[3.3.7 矩阵转置 5](#_Toc31791)

[3.3.8 矩阵求范数 5](#_Toc26807)

[3.4 结构设计图 6](#_Toc9016)

[4 详细设计 7](#_Toc1307)

[4.1 程序流程图 7](#_Toc13870)

[4.1.1 创建稀疏矩阵 7](#_Toc15851)

[4.1.2 打印稀疏矩阵 8](#_Toc14404)

[4.1.3 矩阵加法 10](#_Toc11581)

[4.1.4 矩阵减法 11](#_Toc6197)

[4.1.5 矩阵乘法 13](#_Toc22487)

[4.1.6 矩阵数乘 15](#_Toc15552)

[4.1.7 矩阵转置 16](#_Toc29253)

[4.1.8 矩阵求范数 17](#_Toc30183)

[4.1.9 主程序 18](#_Toc19358)

[5 调试与测试 19](#_Toc278)

[5.1 测试用例设计表 19](#_Toc29573)

[5.2 测试结果 21](#_Toc32678)

[6 运行结果 25](#_Toc7465)

[7 心得与体会 27](#_Toc31320)

# 1 设计题目

题目名称：稀疏矩阵的十字链表法运算模块设计和实现

要求：使用C/C++实现，需要最少实现下标赋值，下标取值，矩阵转置，矩阵的加、减、乘，数乘、范数、转为三元组导出，从三元组文件导入矩阵等运算，以类的形式实现，给出使用的例子。

# 2 设计要求

运用计算机科学与技术专业知识与综合技能，分析与解决工程问题。通过学习、研究与实践使得专业理论深化、知识领域扩展、专业技能延伸。

独立完成题目给定的设计任务，并编写符合要求的设计说明书。

设计内容以软件工程的方法和工具，可以使用UML或软件工程中数据流图、E-R图，流程图等完成设计、编码、测试等工作。

原则上，不能贴代码代替设计中的流程图和功能框图，需要把代码转换为流程图，重要且无法使用流程图的定义，需要加注释说明。

# 3 总体设计

## 3.1 运行环境

运行系统：window 10

运行平台：visual studio 2017

运行内存：1MB

## 3.2 数据来源

本程序数据输入来源于用户键盘输入。

## 3.3 模块设计

### 3.3.1 创建稀疏矩阵

|  |  |
| --- | --- |
| 模块名称 | 创建稀疏矩阵 |
| 模块代号 | 1 |
| 模块功能 | 根据用户输入用十字链表法创建一个稀疏矩阵 |
| 输入 | 矩阵行数，矩阵列数，矩阵非零值个数，矩阵非零值 |
| 输出 | 无 |
| 调用 | 无 |
| 被调用 | 矩阵加法，矩阵减法，矩阵乘法，矩阵数乘，矩阵转置，矩阵求范数 |

### 3.3.2 打印稀疏矩阵

|  |  |
| --- | --- |
| 模块名称 | 打印稀疏矩阵 |
| 模块代号 | 2 |
| 模块功能 | 将稀疏矩阵以三元组（行，列，值）的形式打印出来 |
| 输入 | 稀疏矩阵 |
| 输出 | （行，列，值） |
| 调用 | 无 |
| 被调用 | 矩阵加法，矩阵减法，矩阵乘法，矩阵数乘，矩阵转置，矩阵求范数 |

### 3.3.3 矩阵加法

|  |  |
| --- | --- |
| 模块名称 | 矩阵加法 |
| 模块代号 | 3 |
| 模块功能 | 当两个矩阵行列值相同时，对其进行加法运算，并将相加的结果存储到result矩阵中 |
| 输入 | 矩阵1，矩阵2，矩阵result |
| 输出 | 两个矩阵的相加结果矩阵result |
| 调用 | 创建稀疏矩阵，打印稀疏矩阵 |
| 被调用 | 无 |

### 3.3.4 矩阵减法

|  |  |
| --- | --- |
| 模块名称 | 矩阵减法 |
| 模块代号 | 4 |
| 模块功能 | 当两个矩阵行列值相同时，用第一个矩阵减去第二个矩阵，并将结果存储到result矩阵中 |
| 输入 | 矩阵1，矩阵2，矩阵result |
| 输出 | 两个矩阵的相减结果矩阵result |
| 调用 | 创建稀疏矩阵，打印稀疏矩阵 |
| 被调用 | 无 |

### 3.3.5 矩阵乘法

|  |  |
| --- | --- |
| 模块名称 | 矩阵乘法 |
| 模块代号 | 5 |
| 模块功能 | 当第一个矩阵的列值等于第二个矩阵的行值时，对这两个矩阵进行相乘运算，并将结果存在result矩阵中 |
| 输入 | 矩阵1，矩阵2，矩阵result |
| 输出 | 两个矩阵的相乘结果矩阵result |
| 调用 | 创建稀疏矩阵，打印稀疏矩阵 |
| 被调用 | 无 |

### 3.3.6 矩阵数乘

|  |  |
| --- | --- |
| 模块名称 | 矩阵数乘 |
| 模块代号 | 8 |
| 模块功能 | 用数字与矩阵的每一项相乘，并将结果储存到result矩阵中 |
| 输入 | 矩阵1，数字 |
| 输出 | 矩阵1的数乘结果矩阵result |
| 调用 | 创建稀疏矩阵，打印稀疏矩阵 |
| 被调用 | 无 |

### 3.3.7 矩阵转置

|  |  |
| --- | --- |
| 模块名称 | 矩阵转置 |
| 模块代号 | 7 |
| 模块功能 | 调换矩阵1的行列号，保持其值不变，将其存储在矩阵2中 |
| 输入 | 矩阵1，矩阵2 |
| 输出 | 矩阵2 |
| 调用 | 创建稀疏矩阵，打印稀疏矩阵 |
| 被调用 | 无 |

### 3.3.8 矩阵求范数

|  |  |
| --- | --- |
| 模块名称 | 矩阵求范数 |
| 模块代号 | 8 |
| 模块功能 | 求稀疏矩阵的L0（非0元素的个数）,L1（各个元素的绝对值之和）,L2（所有元素的平方之和然后开跟）范数 |
| 输入 | 矩阵1 |
| 输出 | L0,L1,L2 |
| 调用 | 创建稀疏矩阵，打印稀疏矩阵 |
| 被调用 | 无 |

## 3.4 结构设计图

稀疏矩阵的十字链表设计图

对于稀疏矩阵A，表示如图3-1：

A：

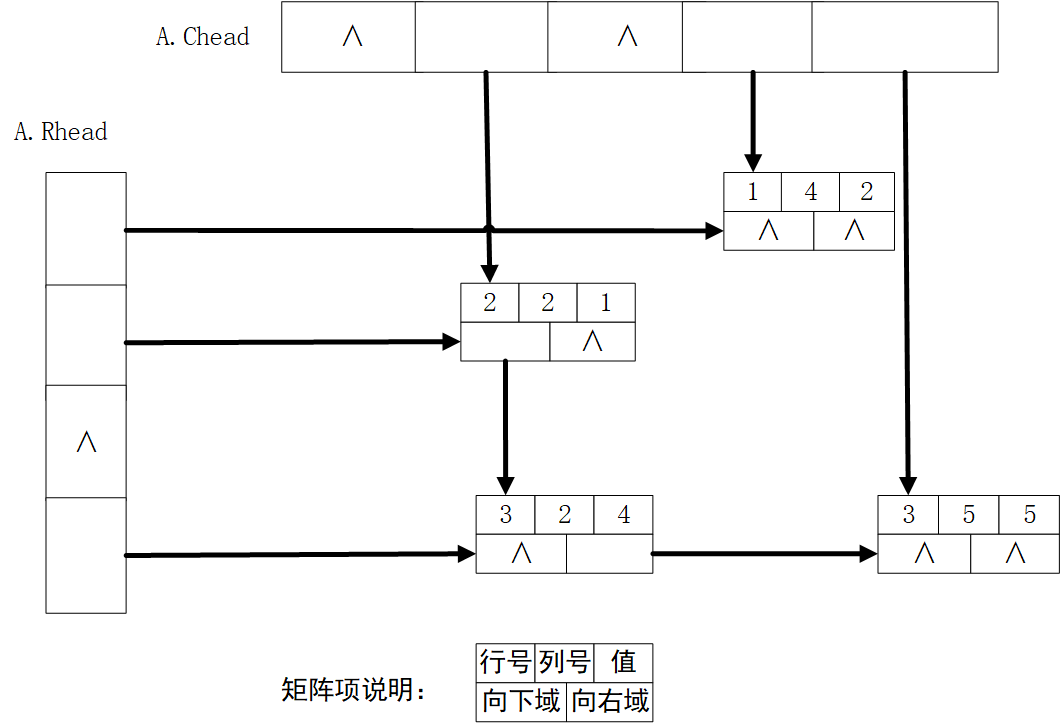


图3-1 十字链表结构图

链表具体结构如图3-2：

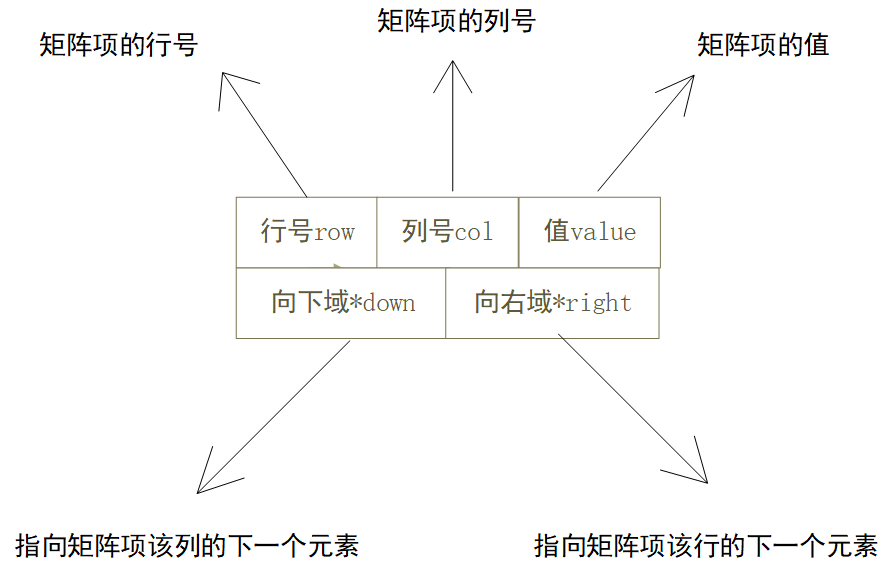


图3-2 链表具体结构图

# 4 详细设计

## 4.1 程序流程图

### 4.1.1 创建稀疏矩阵

流程图如图4-1：

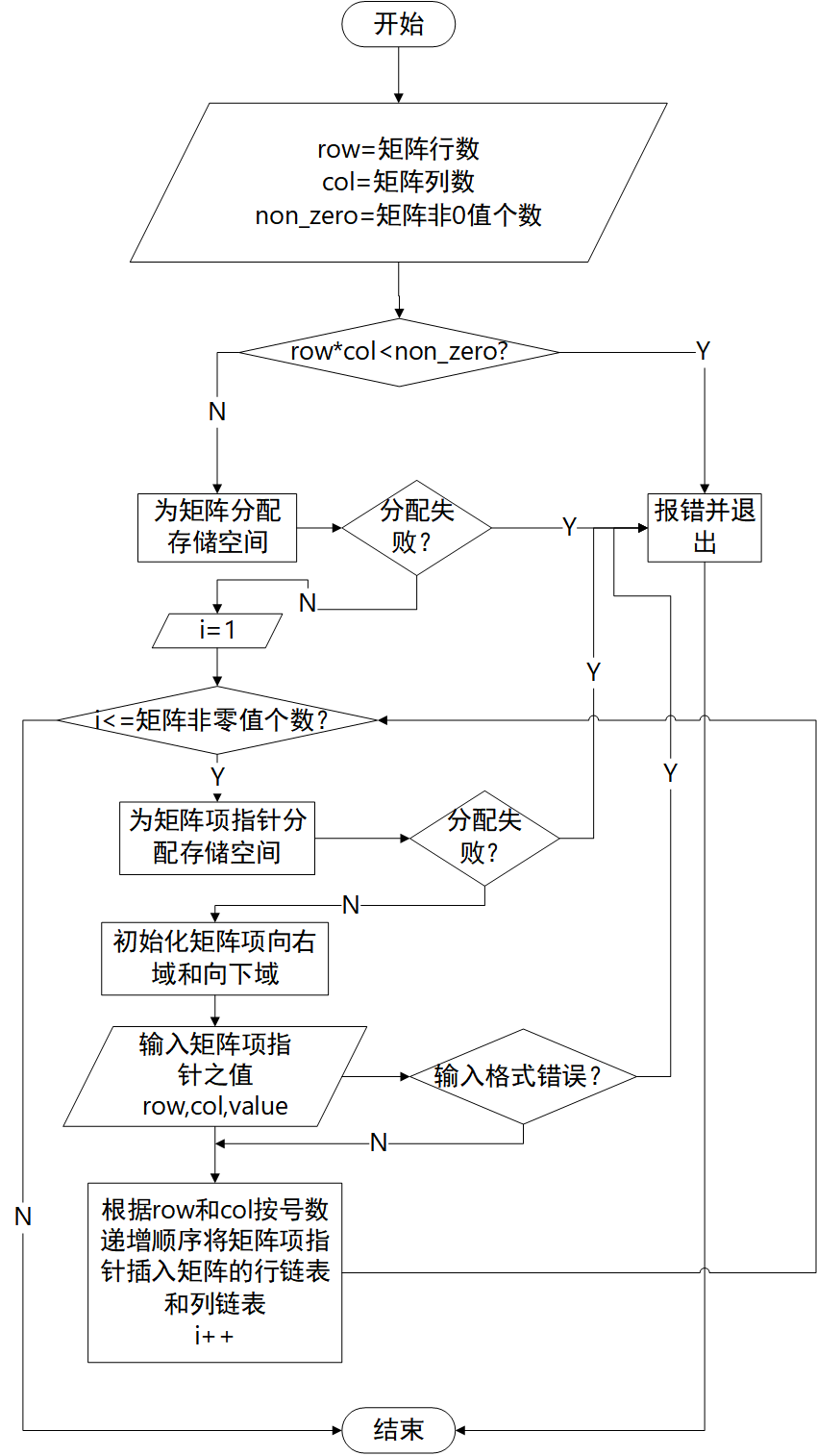


图4-1 创建稀疏矩阵流程图

### 4.1.2 打印稀疏矩阵

流程图如图4-2：

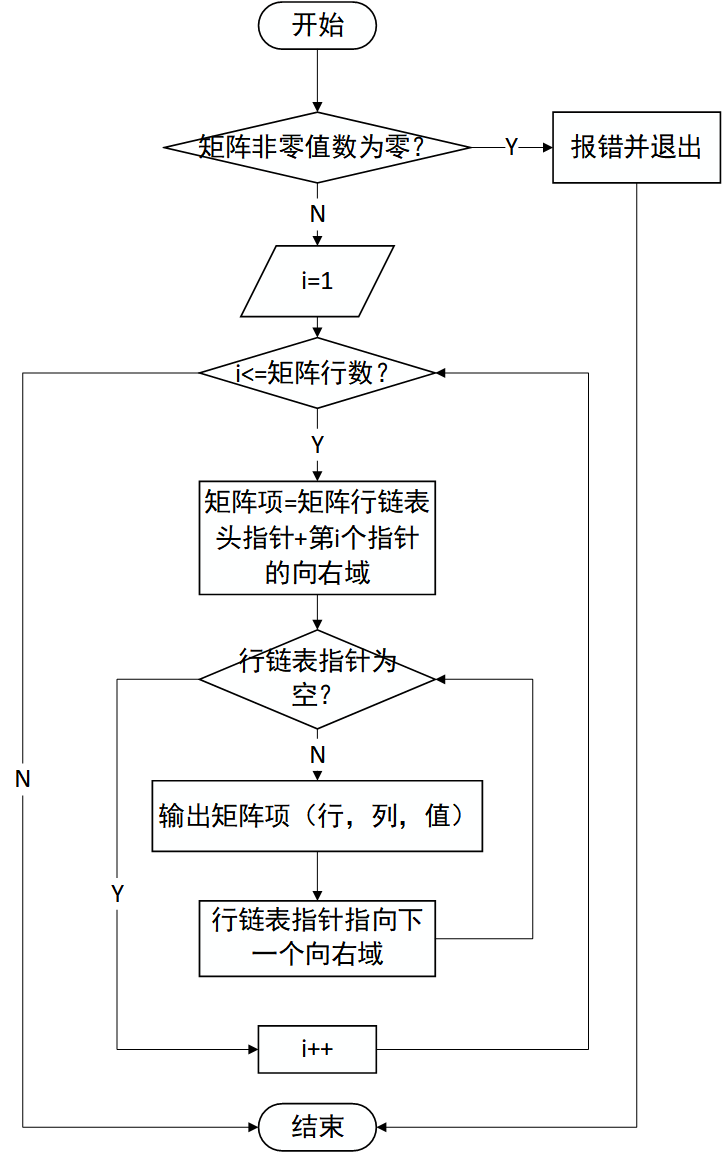


图4-2 打印稀疏矩阵流程图

（注：此空白页是为了打印文档时方便查看分成两页画的流程图而插入）

### 4.1.3 矩阵加法

流程图如图4-3、图4-4：

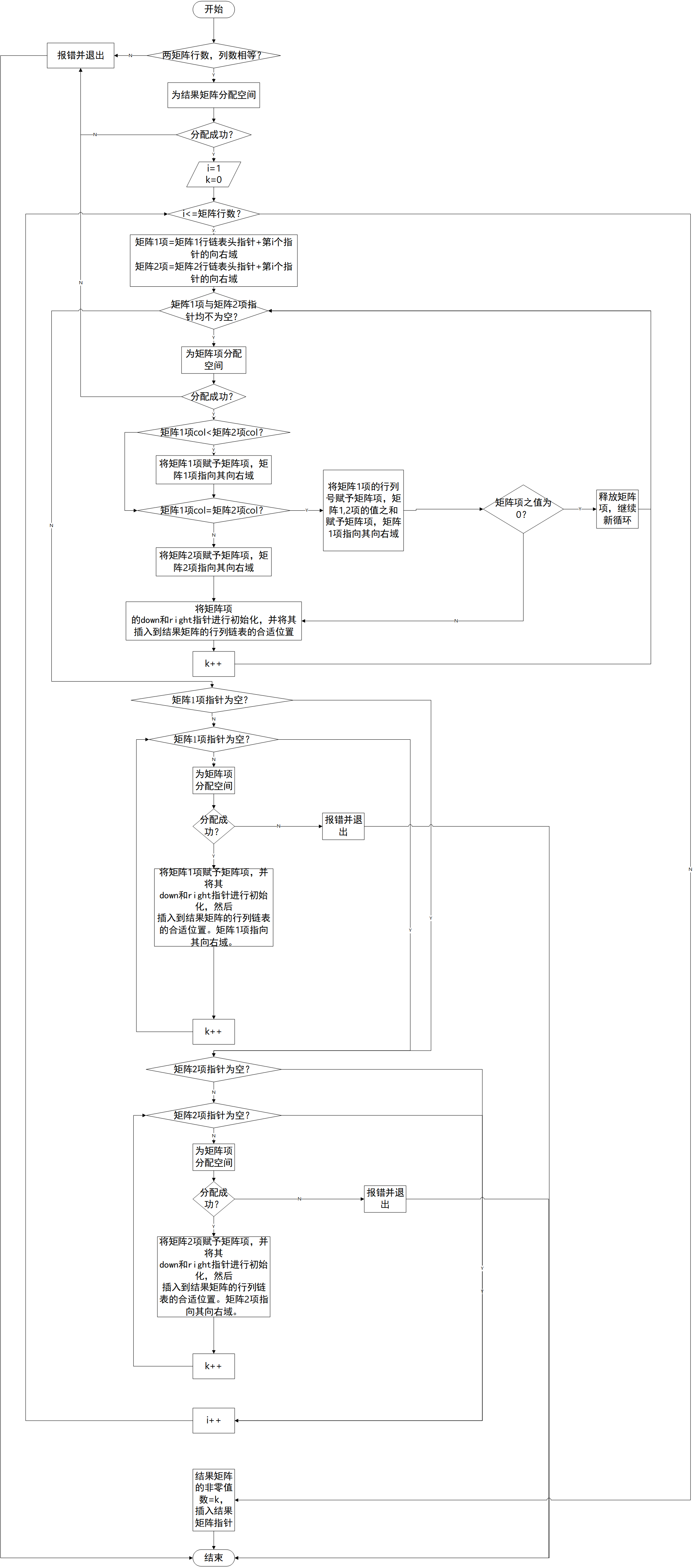


图4-3 矩阵加法流程图上半部分

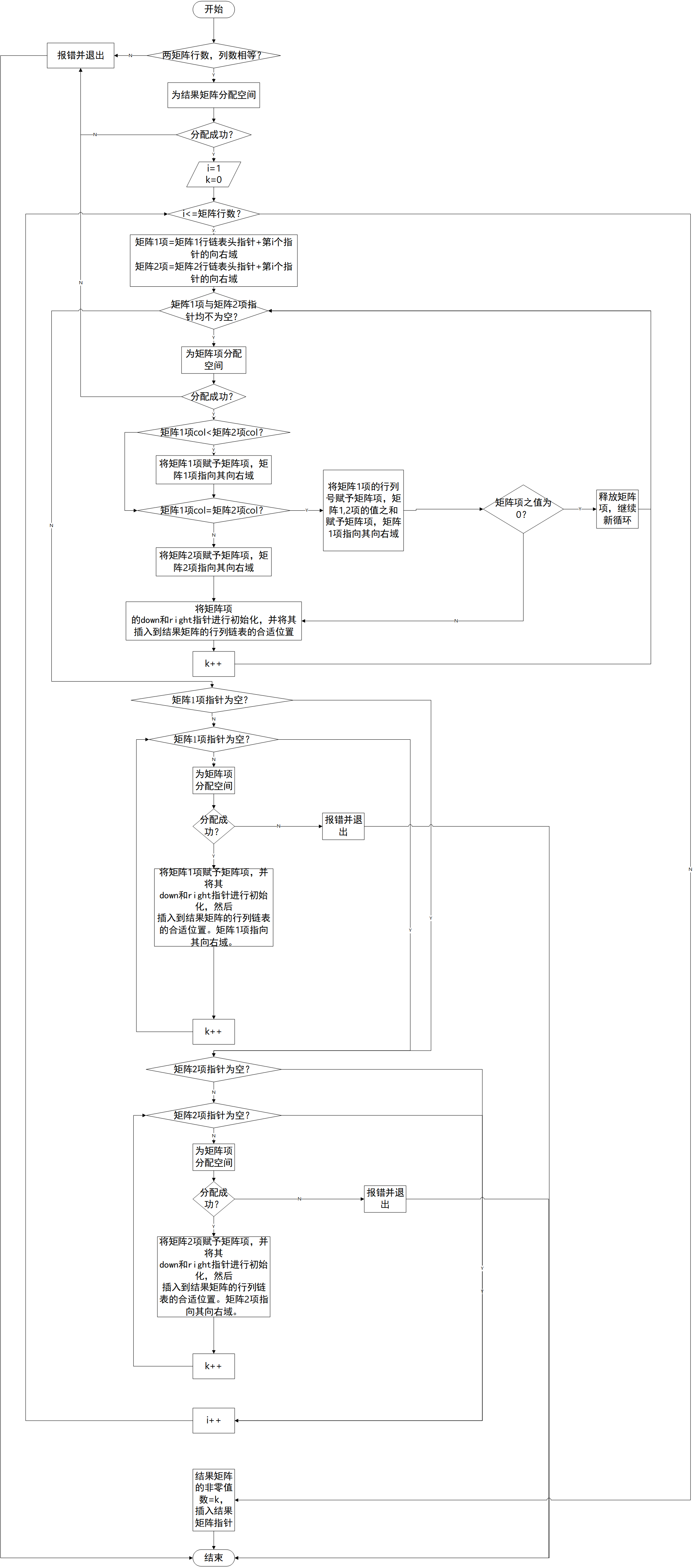


图4-4 矩阵加法流程图下半部分

### 4.1.4 矩阵减法

流程图如图4-5、图4-6：

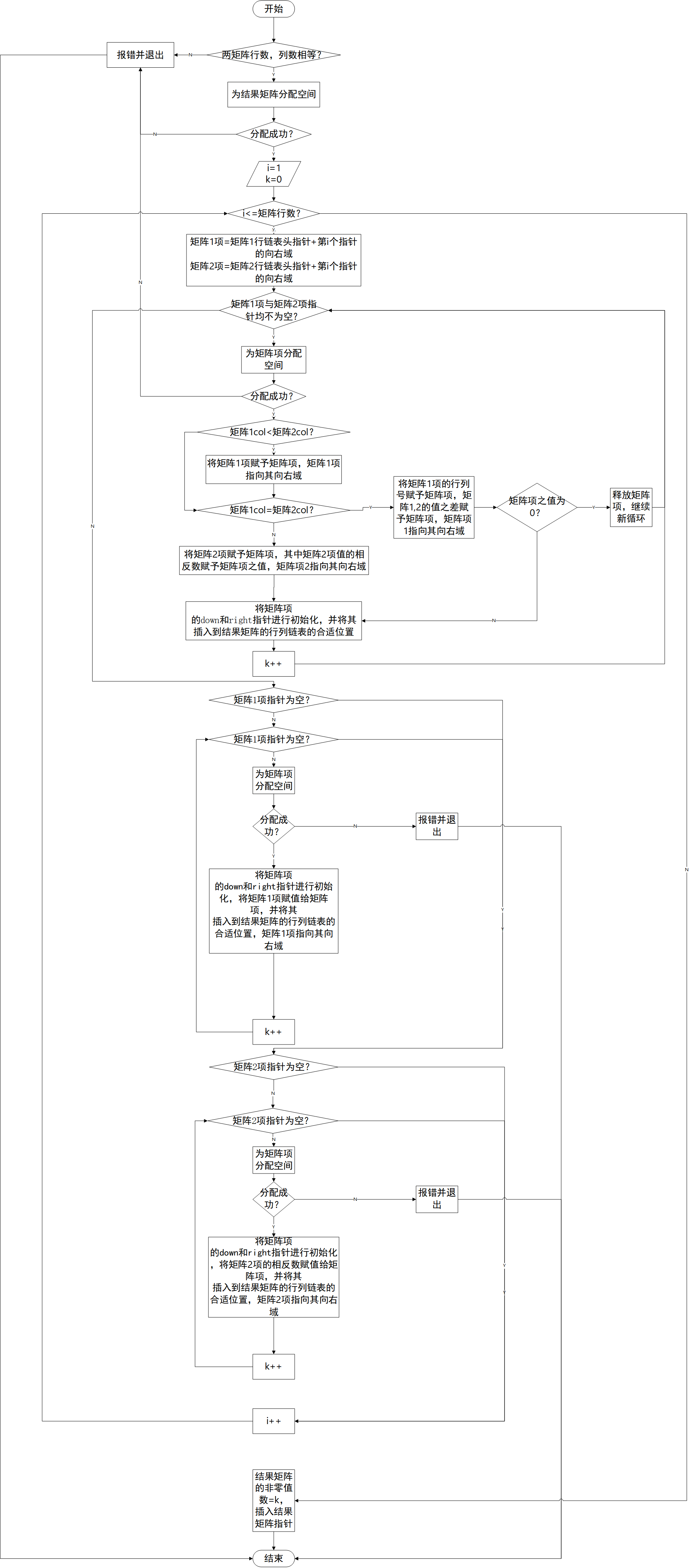


图4-5 矩阵减法流程图上半部分

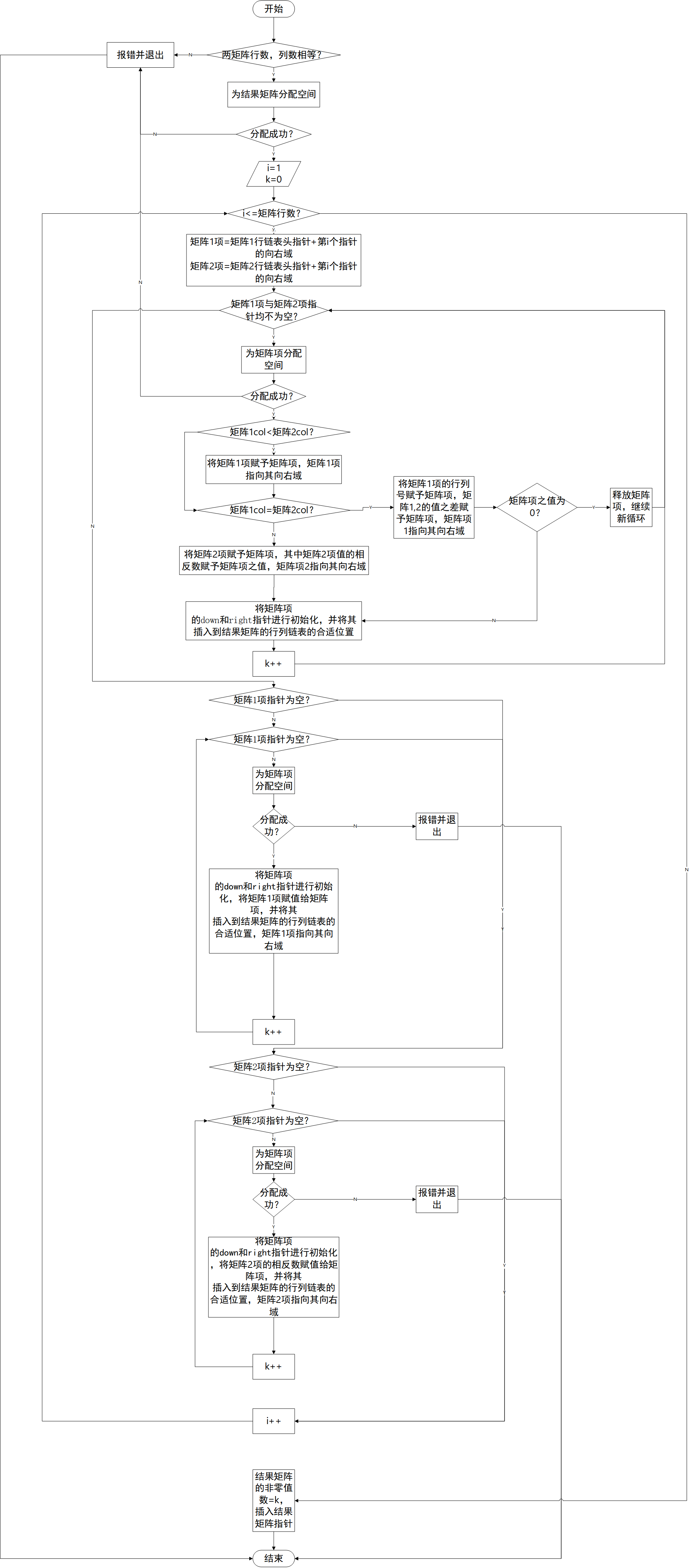


图4-6 矩阵减法流程图下半部分

### 4.1.5 矩阵乘法

流程图如图4-7：

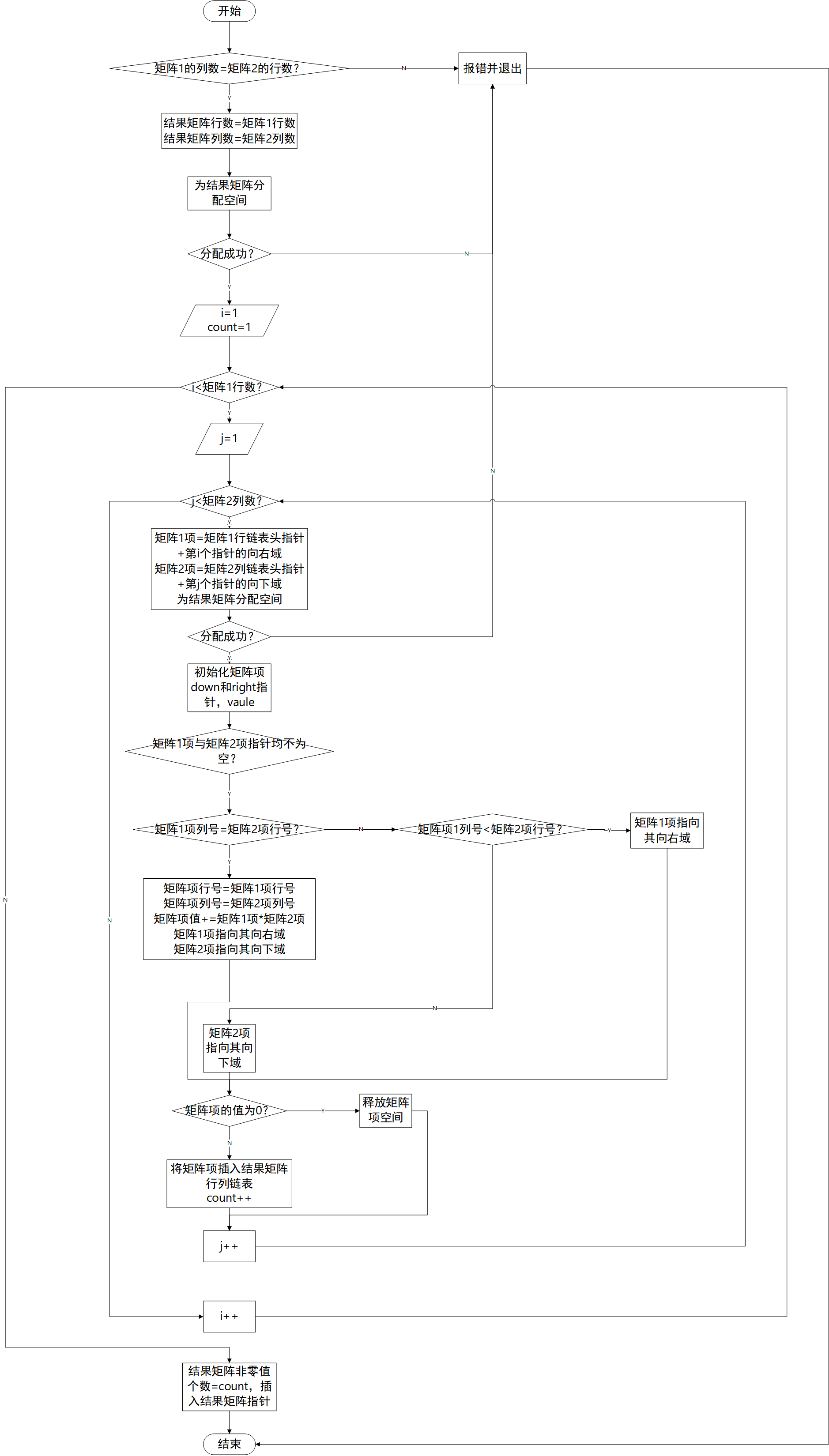


图4-7 矩阵乘法流程图

### 4.1.6 矩阵数乘

流程图如图4-8：

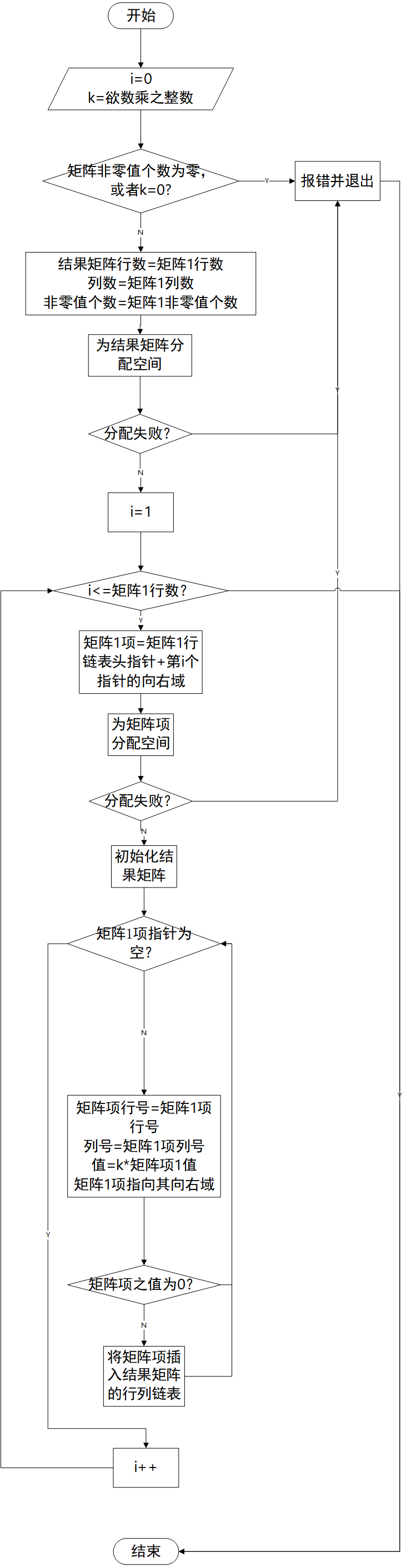


图4-8 矩阵数乘流程图

### 4.1.7 矩阵转置

流程图如图4-9：

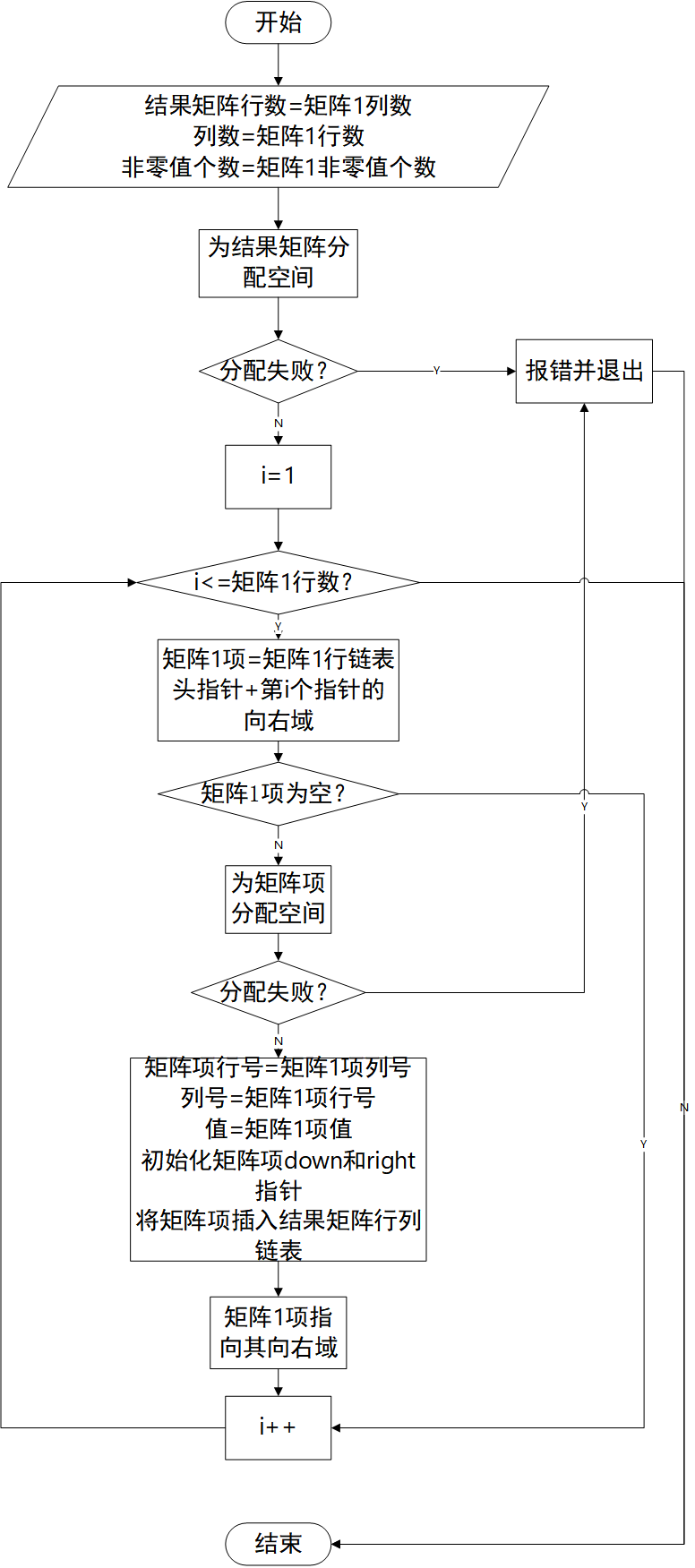


图4-9 矩阵转置流程图

### 4.1.8 矩阵求范数

流程图如图4-10：

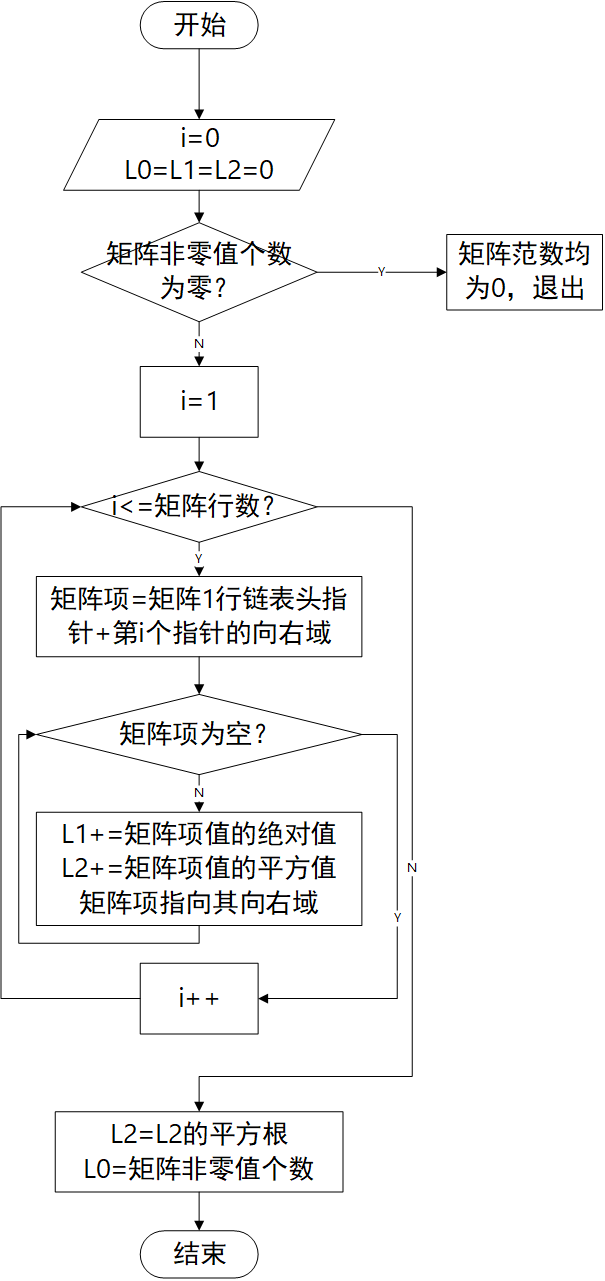


图4-10 矩阵求范数流程图

### 4.1.9 主程序

流程图如图4-11：

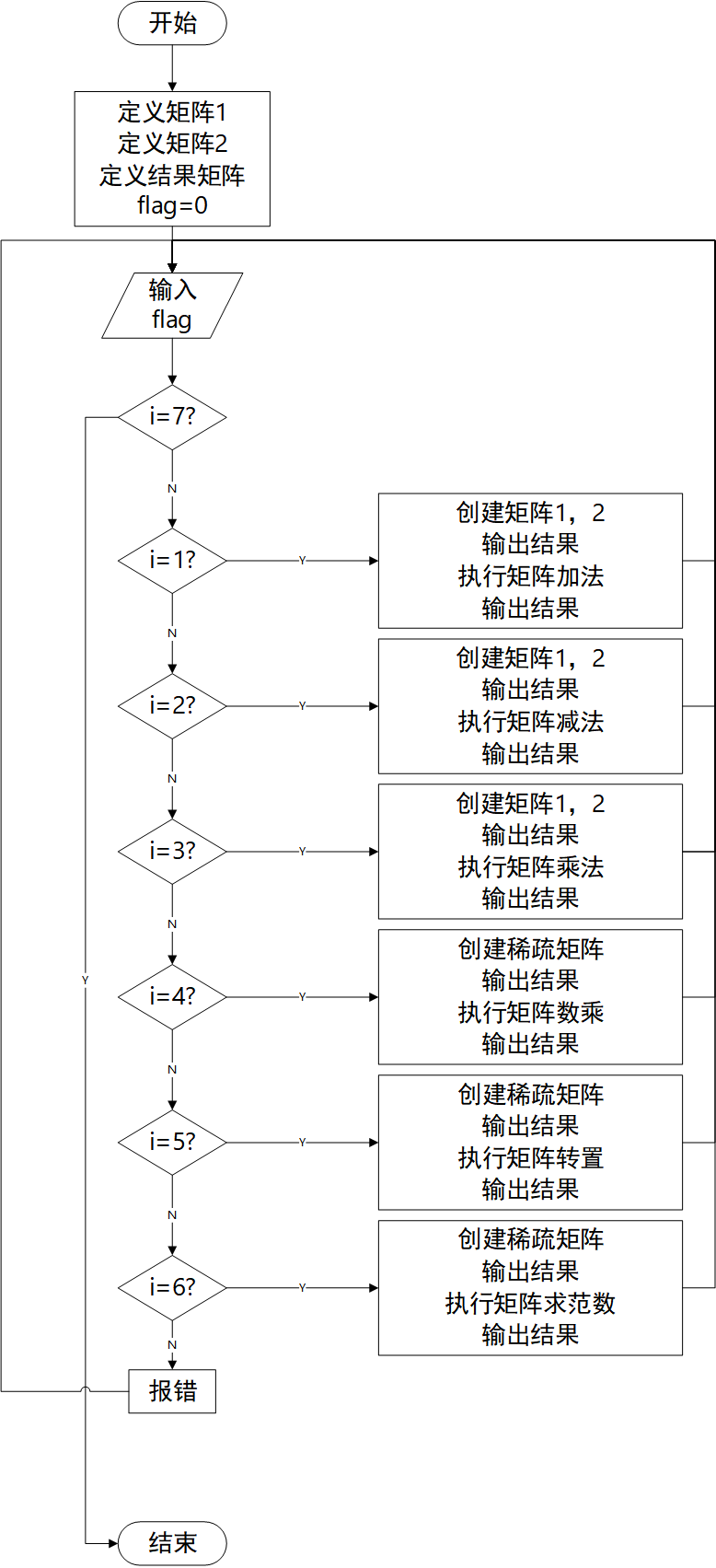


图4-11 主程序流程图

# 5 调试与测试

## 5.1 测试用例设计表

|  |  |  |  |
| --- | --- | --- | --- |
|  | 输入数据 | 预期输出 | 实际输出 |
| 测试用例1  （矩阵加法） | flag=1  j1\_row=3  j1\_col=3  j1\_non\_zero=3  j1非零值：  (1,2,1)  (2,3,5)  (3,1,4)  j2\_row=3  j2\_col=3  j2\_non\_zero=3  j2非零值：  (1,3,3)  (2,3,4)  (3,1,-1) | (1,2,1)  (1,3,3)  (2,3,9)  (3,1,3) | 图5-1 |
| 测试用例2  （矩阵减法） | flag=2  j1\_row=3  j1\_col=3  j1\_non\_zero=3  j1非零值：  (1,2,1)  (2,3,5)  (3,1,4)  j2\_row=3  j2\_col=3  j2\_non\_zero=3  j2非零值：  (1,3,3)  (2,3,4)  (3,1,-1) | (1,2,1)  (1,3,-3)  (2,3,1)  (3,1,5) | 图5-2 |
| 测试用例3  （矩阵乘法） | flag=3  j1\_row=3  j1\_col=3  j1\_non\_zero=3  j1非零值：  (1,2,1)  (2,3,5)  (3,1,4)  j2\_row=3  j2\_col=3  j2\_non\_zero=3  j2非零值：  (1,3,3)  (2,3,4)  (3,1,-1) | (1,3,4)  (2,1,-5)  (3,3,12) | 图5-3 |
| 测试用例4  （矩阵数乘） | flag=4  j1\_row=3  j1\_col=3  j1\_non\_zero=3  j1非零值：  (1,2,1)  (2,3,5)  (3,1,4)  数乘之数=3 | (1,2,3)  (2,3,25)  (3,1,12) | 图5-4 |
| 测试用例5  （矩阵转置） | flag=5  j1\_row=3  j1\_col=3  j1\_non\_zero=3  j1非零值：  (1,2,1)  (2,3,5)  (3,1,4) | (1,3,4)  (2,1,1)  (3,2,5) | 图5-5 |
| 测试用例6  （矩阵求范数） | flag=6  j1\_row=5  j1\_col=5  j1\_non\_zero=5  j1非零值：  (1,2,1)  (2,3,5)  (3,1,4)  (4,5,8)  (5,2,-2) | L0=5  L1=20  L2=10 | 图5-6 |

## 5.2 测试结果

见下图：

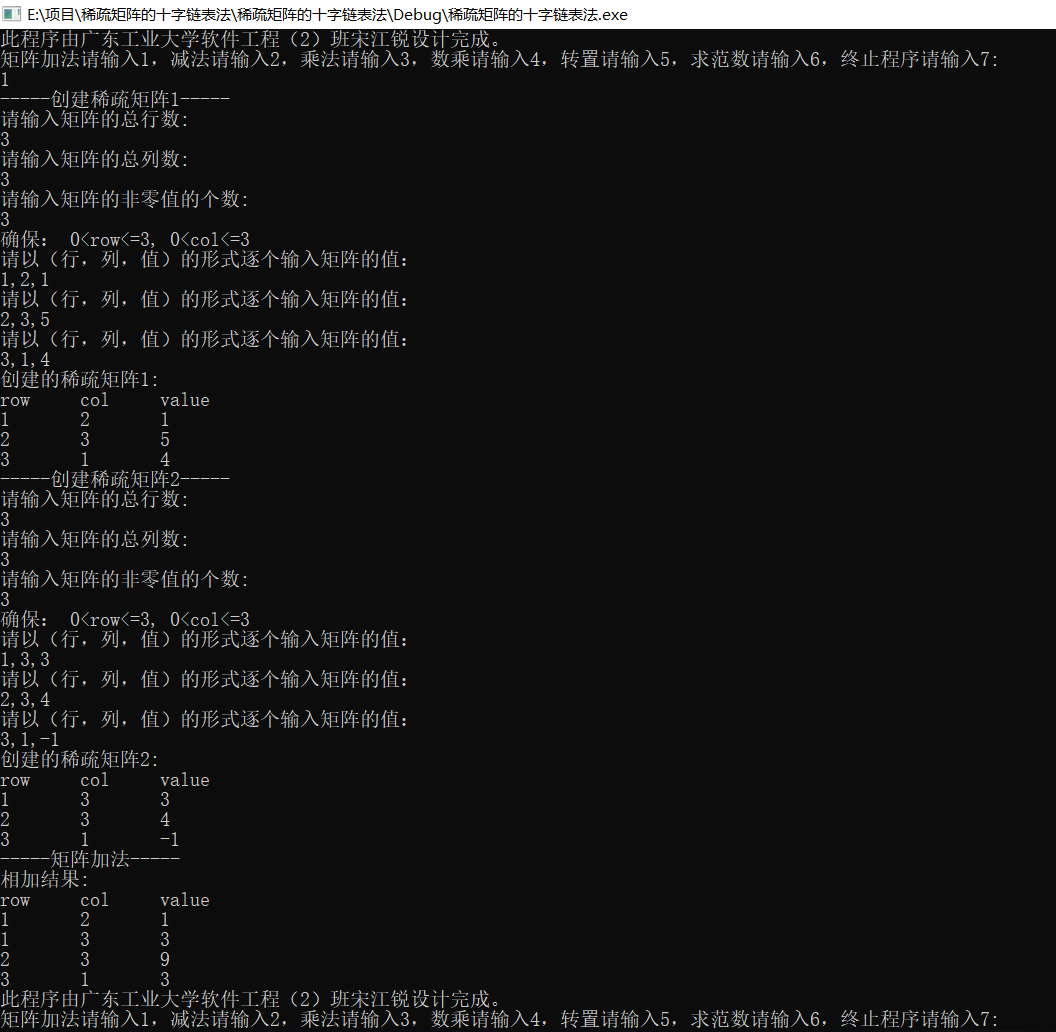


图5-1 矩阵加法测试结果图

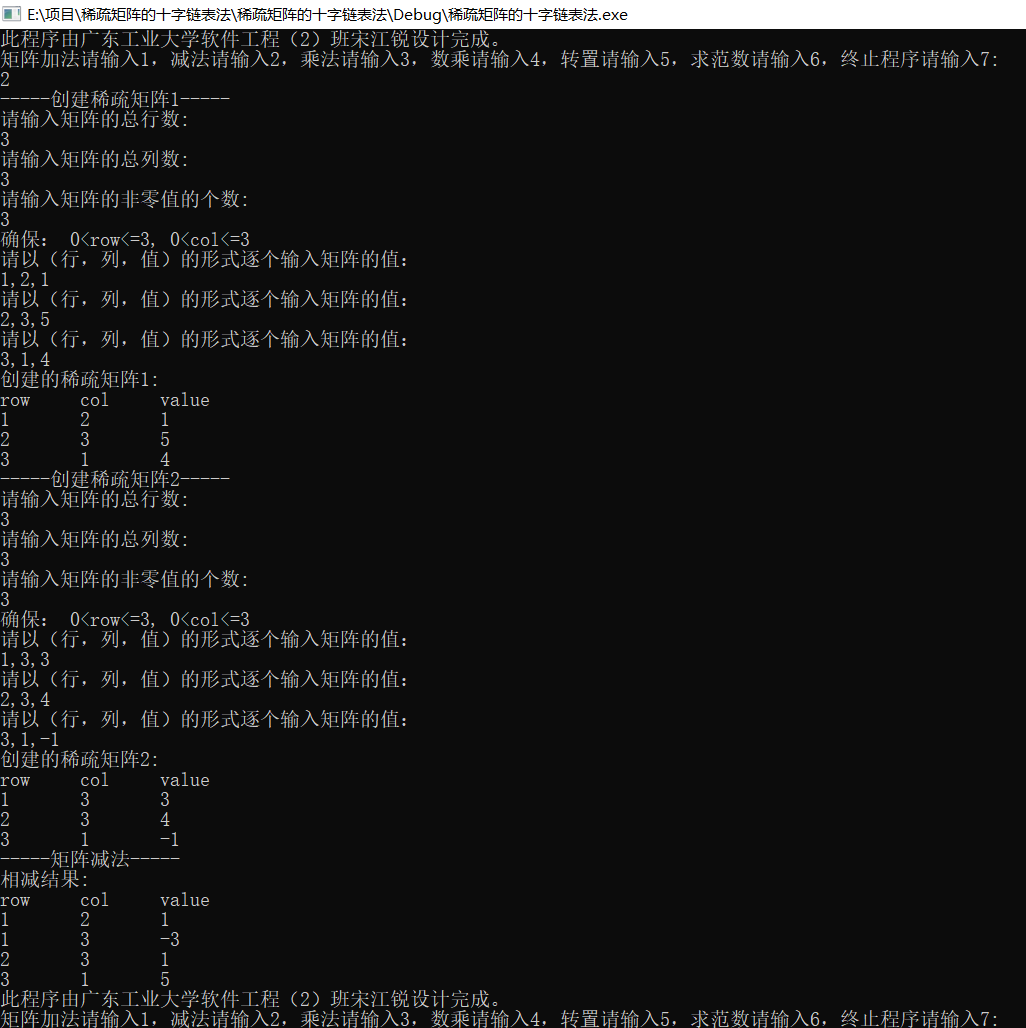


图5-2 矩阵减法测试结果图

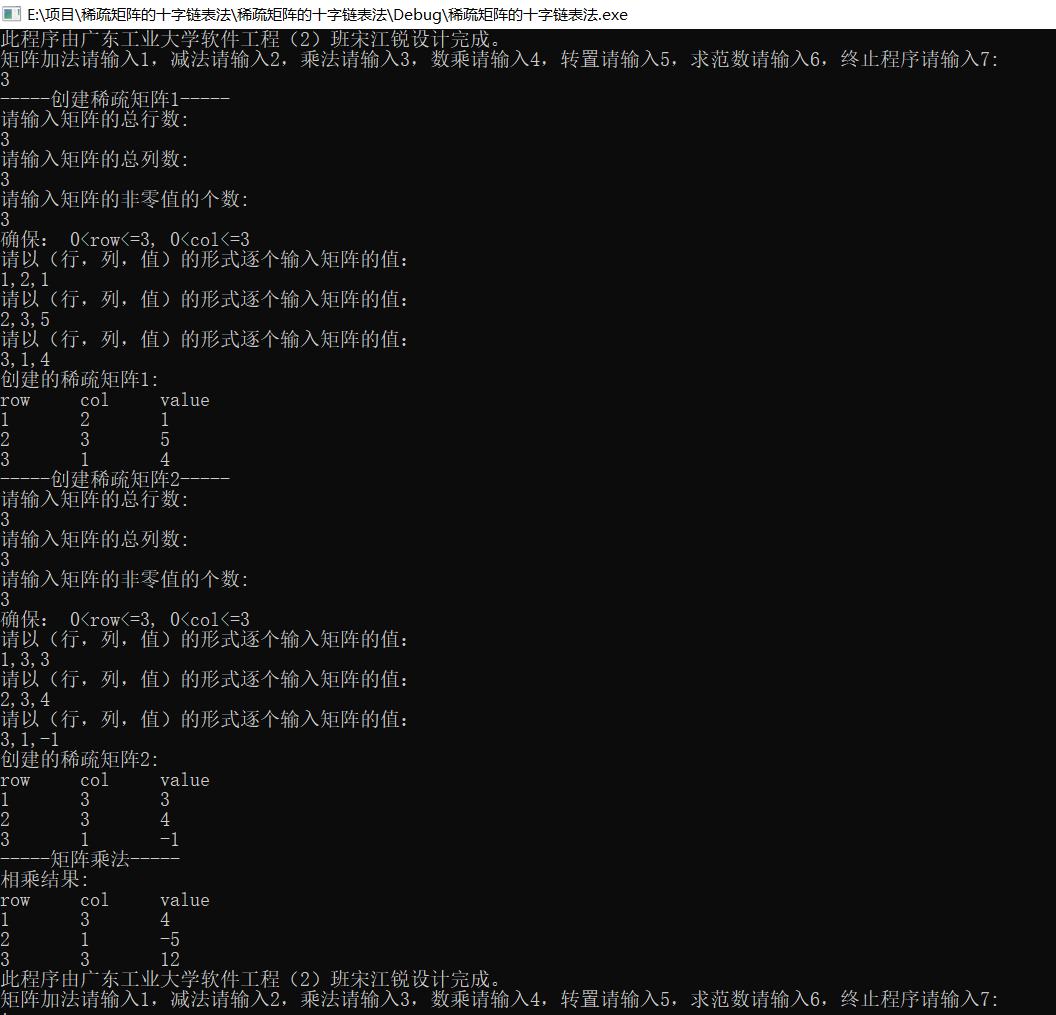


图5-3 矩阵乘法测试结果图

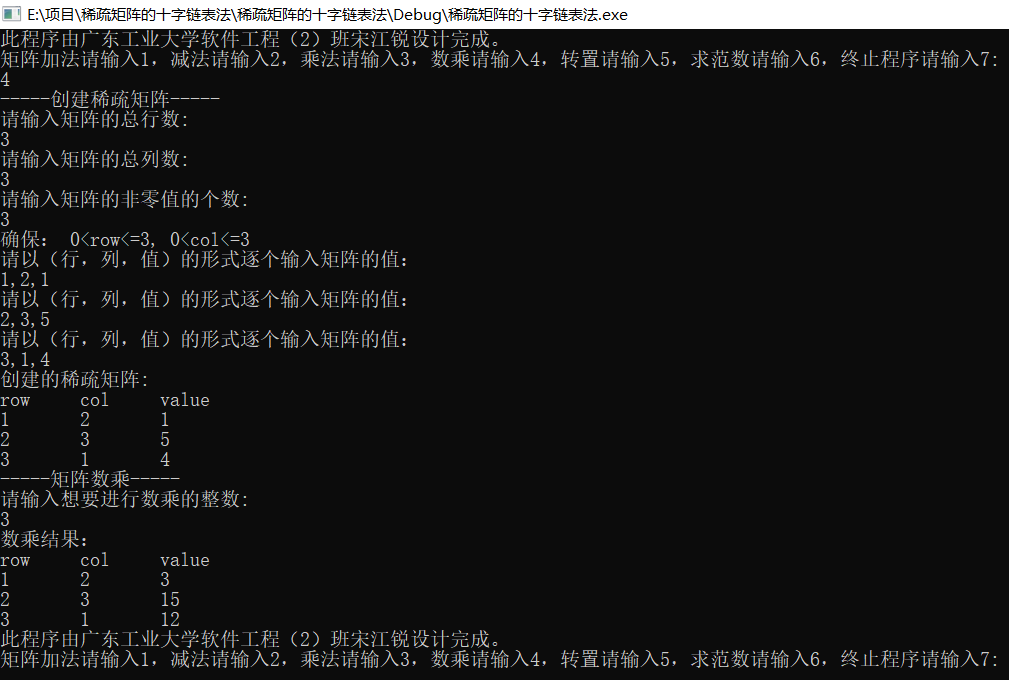


图5-4 矩阵数乘测试结果图

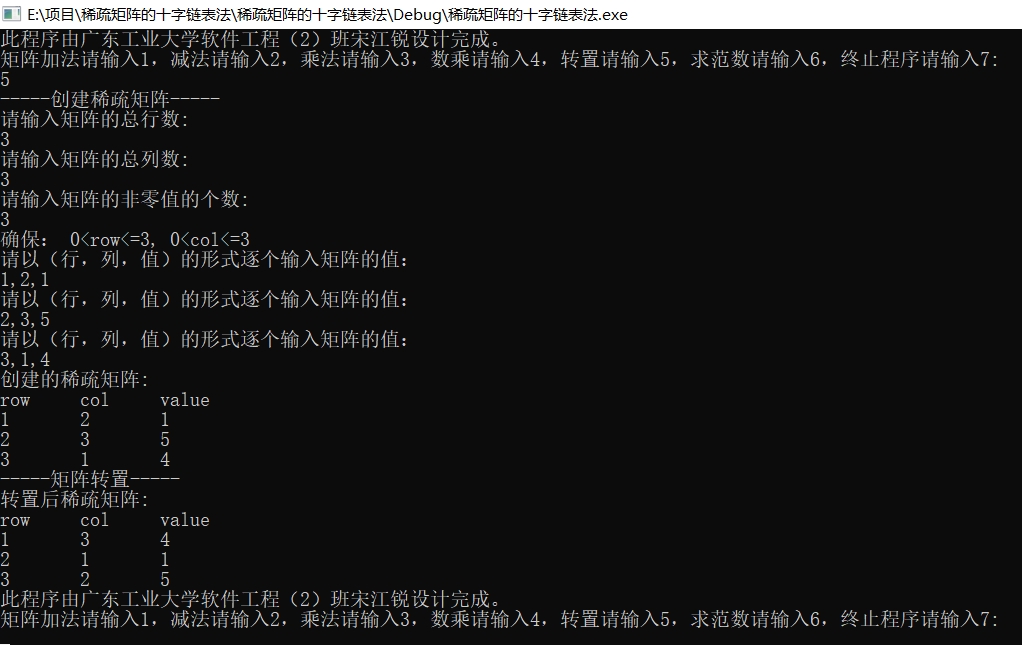


图5-5 矩阵转置测试结果图

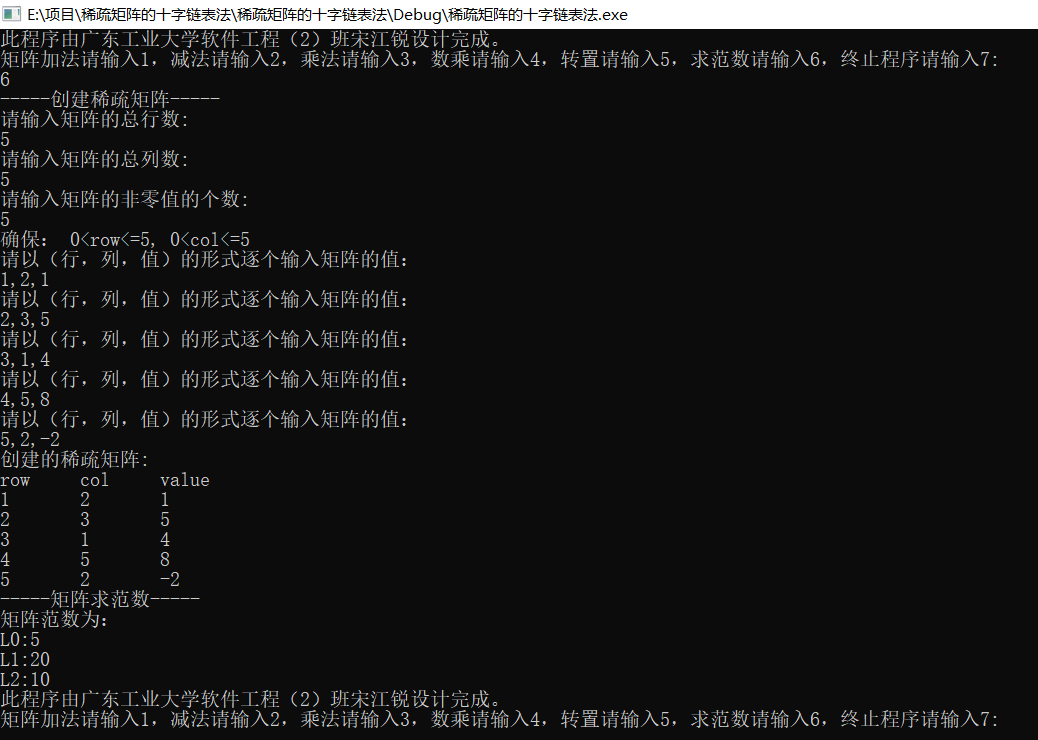


图5-6 矩阵求范数测试结果图

# 6 运行结果

运行结果正常，程序可用。

# 7 心得与体会

在一个月的程序设计时间里，我运用以往的软件工程设计经验，决定先将程序的文档部分完成，然后再动手写代码。在这个过程中，由于过去对于矩阵部分并没有深入的了解，我一边拿起过去的数据结构课本进行复习，一边在网上查找十字链表法的相关结构知识，在遇到难题的时候不断思索，不断尝试改进，最终实现了题目要求的功能。

这一次的设计对我而言相当于踏进了一个崭新的领域，更是对于把实际问题用虚拟数字来描述的方式产生了很大的好奇感，数学建模从来不止一种方式，在积累了深厚的经验和对计算机语言有深刻的理解之后，我们可以用更简单更美妙的方式来描绘虚拟世界。