2016 年 6 月 12 号

第七组

MATRIX CALCULATOR

黄昌俊 肖潼 张建成

目录

Contents

[**Matrix Calculator**]——实验报告

1.课题介绍

1.1 课题背景

众所周知， MATLAB 是一款非常强大的基于矩阵运算的数学软件。但是其庞大的体积、高昂的价格令我们这些只需要最基本矩阵运算功能的大一新生们望而却步。为了解决这个问题，我们小组设计并且编写了这个 Matrix Calculator 来实现基本的矩阵运算功能。

1.2 课题要求

基本功能

分析具体指令

多种方式定义新矩阵

保存计算结果

计算矩阵某行某列的值

计算子矩阵

矩阵的四则运算

矩阵的求和，最大值和最小值

矩阵数值的四舍五入和取整

中级功能

从文件读取指令

向文件写入计算结果

矩阵的幂

矩阵的转置

矩阵的行列式

矩阵的秩

化为最简形

矩阵的逆，除法

向量的内积，模和夹角

命令追溯

高级功能

表达式

特征向量和特征值

命令合法性判断

帮助文档

可选的指令

可以给 MatCalc 加上一些其他的功能。如进制转换、 输出每条指令运算时间、格拉姆-施密特正交化、 各种矩阵分解、 前后缀表达式的计算、 符号表达式的计算、方程根的近似求解、函数图像绘制等等。

数据范围

矩阵的个数<=10^3

矩阵行数、列数<=10^4

矩阵的行数*列数<=10^6 **用户**

1.3 考查知识点

结合线性代数的基础知识和简单算法，综合考察数据处理、结构程序设计、函数程序设计与模块思想、指针与链表、文件输入输出等 C 语言相关知识。

1.4 编译环境

本程序采用 Visual Studio 进行编写。是美国微软公司的开发工具包系列产品。VS 是一个基本完整的开发工具集，它包括了整个软件生命周期中所需要的大部分工具，如 UML 工具、代码管控工具、集成开发环境(IDE)等等。所写的目标代码适用于微软支持的所有平台，包括 Microsoft Windows、Windows Mobile、Windows CE、.NET Framework、.NET Compact Framework 和 Microsoft Silverlight 及 Windows Phone。

2.总体设计

根据 PDF 上的要求，我们小组经过分析后将 Matrix Calculator 分成了输入输出模块，储存模块，指令模块，计算模块，帮助模块和错误反馈模块。接下来大致介绍一下各个模块的分工。

2.1 输入输出模块

输入输出模块实现程序的输入与输出操作。

该模块通过分析储存在全局变量中的指定数据，决定是通过文件读入指令或是通过键盘输入指令。在读取指令后，输入输出模块将指令传递给指令模块进行处理。

另一方面，输入输出模块会根据指令模块传来的命令，调用相应的函数实现矩阵或数据的输出，并通过分析储存在全局变量中的指定数据，决定是否要将结果输出到文件中。

2.2 储存模块

储存模块用来储存矩阵。

在储存模块方面，我们首先将矩阵和矩阵的相关信息，包括矩阵的类型，矩阵的行数和列数构成一个名为 Matrix 的结构体，而后将结构体 Matrix 和矩阵名称以及左右指针打包为一个新的名为 matrix_node 的结构体，用于存取相应的矩阵。

另外储存模块还提供插入矩阵节点，查找矩阵节点，删除矩阵节点，创建普

通矩阵节点，创建临时矩阵节点（保存计算结果），分析矩阵类型，返回和修改矩阵指定位置的值，返回子矩阵等功能。

2.3 指令模块

指令模块用于分析命令的具体含义。

指令模块读入输入输出模块传递过来的指令数据，通过词法分析和语法分析完成对指令的分析，并指定各个模块完成相应的工作。

2.4 计算模块

计算模块保存所有的矩阵计算所需的函数。

计算模块接收指令模块传递过来的指令和参数，选择相应的函数完成对应的计算，并将计算结果保存在储存模块的临时矩阵中，由指令模块决定是否将计算结果保存下来。

2.5 帮助模块

帮助模块处理文件的帮助信息。

当指令模块将数据传递给帮助模块时，帮助模块通过分析指令的名称，输出相应指令的用法和功能。PS：帮助模块的信息不写入文件中。

2.6 错误反馈模块

错误反馈模块用于反馈系统中的各种错误。

以上模块在处理数据时会遇到各种各样的错误，每当遇到错误时，会将相应

的错误代号发送给错误反馈模块，由错误反馈模块指定返回的错误信息。

3.函数设计

3.1 储存模块

3.1.1 Matrix 节点

```
typedef struct matrix_node
{
    Matrix *matrix;
    char* name;
    struct matrix_node *left,
*right;
}Matrix_Node;
```

```
typedef struct
{
    int m,n;
    double **pd;
    long **pl;
}Matrix;
```

Matrix 结构由矩阵的基本信息组成，包括矩阵的行数 m，列数 n，类型 label，以及 long**和 double**两种二级指针。若 label 为 1，则为浮点数矩阵，二级指针**pd 指向矩阵。否则，为整形矩阵，二级指针**pl 指向指针。Matrix_Node 结构体由完整的矩阵节点信息组成，包括矩阵的名称 name，矩阵地址*matrix 以及节点的左右节点地址*left，*right。

3.1.2 新建节点

```
Matrix* stor_create(char* matrix_name, int m, int n, int label);
Matrix* stor_createAns(int label, int m, int n, int label2);
```

stor_create ()函数创建一个 Matrix_Node 节点，该节点 name 值为 matrix_name，创建一个 m*n 的矩阵，矩阵的类型由 label 决定，并将该节点插入二叉树，函数返回创建的节点的矩阵的指针。

stor_createAns()函数创建一个大小为 m*n 的临时矩阵节点 ,该节点的地址为 ans[label1] ,节点的类型由 label2 决定 ,函数返回创建的节点的矩阵的指针。

3.1.3 查找指定节点

```
Matrix* stor_matrix(char* matrix_name);
```

该函数通过扫描矩阵节点构成的二叉树，查找名为 matrix_name 的矩阵节点，函数返回矩阵节点或是 NULL 值。

3.1.4 返回矩阵指定位置

```
void* stor_entry(Matrix *p, int m, int n);

void* stor_ansEntry(int label, int m, int n);
```

以上两个函数返回指定矩阵第 m 行第 n 列的数据，方便对单一位置的数据的读写操作。

3.1.5 分析矩阵类型

```
int stor_type(Matrix *p);
```

该函数通过读取 Matrix 结构中*pd 和*pl 指针的数值，判断矩阵是哪种类型，并返回代表矩阵类型的整数（1 代表浮点型，0 代表整型）。

3.1.6 计算矩阵

```
Matrix* stor_assign(Matrix *dest, Matrix *sour);
Matrix* stor_subMatrix(Matrix *sour, int m, int n, int *row, int *colum);
Matrix* stor_assignSubMatrix(Matrix *dest, Matrix *sour, int m, int n, int *row, int *colum);
```

以上三个函数组合在一起，完成子矩阵元素的提取，合并以及返回。

3.2 指令模块

3.2.1 判断符号

```
static int isPunctuation(char ch);
```

isPunctuation()函数判断一个字符是不是符号。

3.2.2 单词扫描

```
static word scanner(int k)；
```

该函数分析句子中出现的各个单词，并给每个合理的单词分配一个标签

3.2.3 规范格式

```
static int format1(void)；
static int format2(void)；
static int format3(void)；
```

以上三个函数用于去掉方括号外所有空格，将多个空格合并为一个空格，删除多余的符号，将.和整数合并成实数未完成，将.和^合并成新运算符。将命令行规范化，变成方便程序能够识别的格式。

3.2.4 将字转变为词

```
static int letter2word(void)；
```

该函数将单个的字符合成具有实际意义的词。

3.2.5 词法分析

```
static int parser(void)；
```

该函数根据词法进行相应的计算和判断，分析词法的具体含义。

3.2.6 语法翻译

```
int com_interpret(void)；
```

该函数对处理后的命令进行分析，分析命令的具体含义。

3.3 计算模块

3.3.1 矩阵加法

```
void add_Matrix(Matrix* p1, Matrix* p2)；
```

该函数计算两个矩阵的加法，并将结果储存在 ans 临时矩阵节点。

3.3.2 矩阵减法

```
void sub_Matrix(Matrix* p1, Matrix* p2)；
```

该函数计算两个矩阵的减法，并将结果储存在 ans 临时矩阵节点。

3.3.3 矩阵乘法

```
void mul_Matrix(Matrix* p1, Matrix* p2)；
```

该函数计算两个矩阵的乘法，并将结果储存在 ans 临时矩阵节点。

3.3.4 矩阵求逆

```
int inverse_Matrix(Matrix* p);
```

该函数计算矩阵的逆，并将结果储存在 ans 临时矩阵节点。若矩阵能求逆，函数返回 1，否则返回 0。

3.3.5 矩阵除法

```
void div_Matrix(Matrix* p1, Matrix* p2);
```

该函数计算两个矩阵的除法，并将结果储存在 ans 临时矩阵节点。

3.3.6 矩阵的幂

```
void ex_Matrix(Matrix* p, int n);
```

该函数计算矩阵的幂，并将结果储存在 ans 临时矩阵节点。

3.3.7 矩阵的元素和

```
double sum_Matrix(Matrix* p);
```

该函数计算矩阵的元素之和，并返回所求值。

3.3.8 矩阵的最大值

```
double max_Matrix(Matrix* p);
```

该函数计算矩阵的最大元素，并返回所求值。

3.3.9 矩阵的最小值

```
double min_Matrix(Matrix* p);
```

该函数计算矩阵的最小元素，并返回所求值。

3.3.10 矩阵四舍五入

```
void round_Matrix(Matrix* p);
```

该函数将矩阵的每个元素都四舍五入，并将结果储存在 ans 临时矩阵节点。

3.3.11 矩阵向上取整

```
void upper_Matrix(Matrix* p);
```

该函数将矩阵的每个元素都向上取整，并将结果储存在 ans 临时矩阵节点。

3.3.12 矩阵向下求整

```
void lower_Matrix(Matrix* p);
```

该函数将矩阵的每个元素都向下取整，并将结果储存在 ans 临时矩阵节点。

3.3.13 矩阵转置

```
void reverse_Matrix(Matrix* p);
```

该函数将矩阵进行转置，并将结果储存在 ans 临时矩阵节点。

3.3.14 矩阵向下求整

```
double det_Matrix(Matrix* p);
```

该函数求矩阵的行列式，并返回计算结果。

3.3.15 矩阵的摸

```
double norm_Matrix(Matrix* p);
```

该函数求矩阵的模，并返回计算结果。

3.3.16 矩阵的秩

```
int rank_Matrix(Matrix* p);
```

该函数求矩阵的秩，并返回计算结果。

3.3.17 矩阵的内积

```
double dot_Matrix(Matrix* p1, Matrix* p2);
```

该函数求矩阵的内积，并返回计算结果。

3.3.18 矩阵的夹角

```
double angle_Matrix(Matrix* p1, Matrix* p2);
```

该函数求矩阵的夹角，并返回计算结果。

3.3.19 矩阵的最简式

```
void rref_Matrix(Matrix* p);
```

该函数求矩阵的最简式，并将结果储存在 ans 临时矩阵节点。

3.3.20 矩阵的特征向量

```
void feature_vector(Matrix* p);
```

该函数求矩阵的特征向量，并将结果储存在 ans 临时矩阵节点。

3.3.21 矩阵的特征值

```
void feature_value(Matrix* p);
```

该函数求矩阵的特征值，并将结果储存在 ans 临时矩阵节点。

3.3.22 矩阵元素的倒数

```
void opposite(Matrix* p);
```

该函数求矩阵的元素的倒数，并将结果储存在 ans 临时矩阵节点。

3.4 错误模块

3.4.1 返回错误信息

```
char* get_error(int label);
```

该函数根据输入的参数，返回错误信息。

3.5 帮助模块

3.5.1 返回帮助信息

```
char* get_help(char *cmd);
```

该函数根据输入的字符串，判断对应的命令，返回相应的帮助信息。

3.5 帮助模块

3.5.1 返回帮助信息

```
char* get_help(char *cmd);
```

该函数根据输入的字符串，判断对应的命令，返回相应的帮助信息。

4.测试结果

很惭愧，我们并没能在规定时间内完成大程，所以也无从谈起测试结果……之后我们彻底完成这个大程后会将相应的测试结果附上。

5.成员分工

讨论：黄昌俊，肖潼，张建成

代码编写：黄昌俊，肖潼，张建成

资料查询：肖潼

文档编辑：黄昌俊