セメスタ課題に関する注意 (再掲)：(他の課題とは違って) セメスタ課題では、作成したプログラムに関するレポートも提出しなければいけません。このレポートには、(実験のレポートのように) 目的、原理、考察を書かなければいけません。また、実験のレポートではあまり書くことがないかもしれませんが、作成物 (プログラム) の使用法も書かなければいけません。(man コマンドで表示されるものを参考にするといいでしょう。)

セメスタ課題 II(行列演算) TS2

(2005/06/16(Thu.))

TS2-1:行列演算 (ソース)
(本提出期限 2005/07/14/(Thu.)17:40、再提出期限 2005/07/28(Thu.)20:00)

提出ファイル:Makefile, ソースファイル (matrix.c), 入力ファイル (matrix.in), 出力ファイル (matrix.out)
正方行列の各種演算を行う行列電卓プログラムを作成せよ。プログラムの仕様は別紙参照。
TS2-2:行列演算 (レポート)
(本提出期限 2005/07/14(Thu.)17:40、再提出期限 2005/07/28(Thu.)(20:00)

提出物:レポート (紙)、あるいはレポートのテキストファイル (matrix.txt)。

TS2-1 のプログラムを説明するレポートを提出する。電子的に提出する場合は次のように行なうこと。

```
~/TS2/2$submit TS2 2 matrix.txt
```

行列演算プログラム仕様 TS2

行列演算は以下の 5 種類の演算を行えるようにせよ。

演算種類

(1) スカラー倍

(2) 転置

(3) 逆行列

(4) 行列の和

(5) 行列の積

プログラムは以下に示す操作手順で利用できるようにせよ。

利用方法

1. 正方行列の次数 (行列の大きさを表す整数)n を標準入力から入力する
2. $n \times n$ の正方行列 (被演算行列) を、これから行う演算に必要な個数だけ標準入力から入力する
3. 演算の種類を表わす記号を標準入力から入力する
4. 演算結果 ($n \times n$ の正方行列) が標準出力に出力される

また、プログラムは次の条件を満足するように作成せよ。

要求仕様

- 被演算行列として、100×100 の正方行列 (100 次正方行列) まで扱えるようにすること。この値はプログラム先頭でマクロ定義し、マクロ定義の部分だけを書き換えるだけで変更できるようにすること。
- 各行列の値は行優先で入力できるようにすること。(ある行の値 n 個を入力した後で、次の行の入力を行う)
- スカラー倍演算を行うときは、手順 3. でスカラー倍を表す記号を入力した後に、スカラー値を入力できるようにすること。
- 行列の各要素やスカラー倍で用いるスカラー値としては任意の実数を入力できるようにすること。
- 演算処理はそれぞれの演算の種類に対し 1 個の関数を作成すること。各関数は次ページのプロトタイプ宣言を持つようにすること。
- 行列演算を行う関数は、行列電卓プログラム以外の他のプログラム中からも使えるようにすること。すなわち、演算を行う関数の中で行列の要素の値や演算を表す記号を標準入力から読み込んだり、演算を行う関数の中で結果の表示を行ってはならない。
- グローバル変数を用いてはならない。

以下の点については各自で自由に設計してよい。

自由設計

+ 演算を表す記号は自由に決めてよい。例えば、
  * スカラー倍:'s'
  * 転置:'t'
  * 逆行列:'i'
  * 行列の和:'a'
  * 行列の積:'p'

  など。

+ 単項演算 (演算に一つの行列しか利用しない演算) を行う場合でも、操作手順の 2. において余分に行列を入力しなくてはならないようにしてもよい。例えば、常に二つの $n \times n$ 正方行列を入力してから演算の種類を表す記号を入力するようにしてもよい。その場合、最初に入力された行列に対して演算を行い、後に入力された行列の値は無視するようにすること。

+ 操作手順の 2. と 3. は逆の順序で行うようにしてもよい。その場合、演算の種類を表す記号を入力してから、その演算に必要な行列を入力することになる。

+ 1 個の $n \times n$ 行列の全ての要素を標準入力から読み取ることのできる関数を 1 個作成し、手順 2. で利用するようにしてもよい (次ページプロトタイプ参考)。

+ 1 個の $n \times n$ 行列の全ての要素を標準出力に出力する関数を 1 個作成し、手順 4. で利用するようにしてもよい (次ページプロトタイプ参考)。

+ 出力の表示の仕方については、各自工夫すること。後に掲載する出力例は参考であり、必ずしもこの通りでなくともよい。

マクロ定義例

```
/* マクロ定義 */
#define N_MAX (100)  /* 扱える正方行列の最大の次数 */

/* プロトタイプ宣言 */

/*
 * 正方行列のスカラー倍演算を行う関数
 *
 * 引数 :
 * n       : 正方行列の次数
 * k       : スカラー値。 スカラー倍演算によって、被演算行列の各要素が k 倍される。
 * operand : 被演算行列 (n 次正方行列)
 * result  : スカラー倍演算結果を格納する場所 (n 次正方行列)
 *
 * 戻り値 : なし
 */
void scalar_multiply(int n, int k,
  double operand[N_MAX][N_MAX],
  double result[N_MAX][N_MAX]);

/*
 * 正方行列の転置演算を行う関数
 *
 * 引数 :
 * n       : 正方行列の次数
 * operand : 被演算行列 (n 次正方行列)
 * result  : スカラー倍演算結果を格納する場所 (n 次正方行列)
 *
 * 戻り値 : なし
 */
void transporse(int n,
  double operand[N_MAX][N_MAX],
  double result[N_MAX][N_MAX]);

/*
 * 正方行列の逆行列演算を行う関数
 *
 * 引数 :
 * n       : 正方行列の次数
 * operand : 被演算行列 (n 次正方行列)
 * result  : 逆行列演算結果を格納する場所 (n 次正方行列) (逆行列が存在する場合)
```

```
 *          : 被演算行列と同一の行列を格納した場所        (逆行列が存在しない場合)
 *
 * 戻り値 : 逆行列が存在する場合は真を、存在しない場合は偽を表す真偽値
 */
int inverse(int n,
  double operand[N_MAX][N_MAX],
  double result[N_MAX][N_MAX]);

/*
 * 正方行列の和演算を行う関数
 *
 * 引数 :
 * n        : 正方行列の次数
 * operand1 : 被演算行列 (和演算の左辺)(n 次正方行列)
 * operand2 : 被演算行列 (和演算の右辺)(n 次正方行列)
 * result   : 和演算結果を格納する場所 (n 次正方行列)
 *
 * 戻り値  : なし
 */
int add(int n,
  double operand1[N_MAX][N_MAX], double operand2[N_MAX][N_MAX],
  double result[N_MAX][N_MAX]);

/*
 * 正方行列の積演算を行う関数
 *
 * 引数 :
 * n        : 正方行列の次数
 * operand1 : 被演算行列 (積演算の左辺)(n 次正方行列)
 * operand2 : 被演算行列 (積演算の右辺)(n 次正方行列)
 * result   : 積演算結果を格納する場所 (n 次正方行列)
 *
 * 戻り値  : なし
 */
int product(int n,
  double operand1[N_MAX][N_MAX], double operand2[N_MAX][N_MAX],
  double result[N_MAX][N_MAX]);

/*
 * 正方行列を標準入力から読み込む関数
 *
 * 引数 :
 * n       : 正方行列の次数
```

```
 * matrix  : 読み込んだ内容(n次正方行列)
 *
 * 戻り値  : なし
 */
void scan_matrix(int n, double matrix[N_MAX][N_MAX]);

/*
 * 正方行列を標準出力へ出力する関数
 *
 * 引数 :
 * n       : 正方行列の次数
 * matrix  : 出力する内容(n次正方行列)
 *
 * 戻り値  : なし
 */
void print_matrix(int n, double matrix[N_MAX][N_MAX]);
```

実行例 1

```
==========================================================================
~TS2/1/$./matrix
正方行列の次数を入力してください:

3

1 個目の 3 次正方行列 A を入力してください:

1.0 0.0 0.0
0.0 2.0 0.0
0.0 0.0 3.0

2 個目の 3 次正方行列 B を入力してください:

0.0 2.0 0.0
0.0 0.0 3.0
4.0 0.0 0.0

演算子の種類を入力してください:
s

スカラー倍演算に使うスカラー値を入力してください:
-3.0

3 次正方行列 A のスカラー倍演算を実行します。

A =
 1.0 0.0 0.0
 0.0 2.0 0.0
 0.0 0.0 3.0


-3.0 A =
 -3.0 -0.0 -0.0
 -0.0 -6.0 -0.0
 -0.0 -0.0 -9.0
~TS2/1/$
==========================================================================
```

入力ファイル (matrix.in) 例

```
-----------------------------------------------------------------
3

1.0 0.0 0.0
0.0 2.0 0.0
0.0 0.0 3.0

0.0 2.0 0.0
0.0 0.0 3.0
4.0 0.0 0.0

s

-3.0
-----------------------------------------------------------------
```

実行例 2(出力 (matrix.out) 例)

```
================================================================================
~TS2/1/$./matrix <matrix.in
正方行列の次数を入力してください:


1 個目の 3 次正方行列 A を入力してください:


2 個目の 3 次正方行列 B を入力してください:

演算子の種類を入力してください:

スカラー倍演算に使うスカラー値を入力してください:

3 次正方行列 A のスカラー倍演算を実行します。

A =
 1.0 0.0 0.0
 0.0 2.0 0.0
 0.0 0.0 3.0


-3.0 A =
 -3.0 -0.0 -0.0
 -0.0 -6.0 -0.0
```

```
 -0.0 -0.0 -9.0
~TS2/1/$
================================================================================
```

実行例 3

```
================================================================================
~TS2/1/$./matrix
正方行列の次数を入力してください:

3

1 個目の 3 次正方行列 A を入力してください:

1.0 0.0 0.0
0.0 2.0 0.0
0.0 0.0 3.0

2 個目の 3 次正方行列 B を入力してください:

0.0 2.0 0.0
0.0 0.0 3.0
4.0 0.0 0.0

演算子の種類を入力してください:
a

3 次正方行列 A と B の和演算を実行します。

A =
 1.0 0.0 0.0
 0.0 2.0 0.0
 0.0 0.0 3.0

B =
 0.0 2.0 0.0
 0.0 0.0 3.0
 4.0 0.0 0.0

A + B =
 1.0 2.0 0.0
 0.0 2.0 3.0
 4.0 0.0 3.0
================================================================================
```