

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define ROW 3 /* 行の要素数 */
5 #define COLUMN 4 /* 列の要素数 */
6 /* 行列の領域確保 */
7 double **dmatrix(int nr1, int nr2, int n11, int n12);
8 /* 行列の領域解放 */
9 void free_dmatrix(double **a, int nr1, int nr2, int n11, int n12);
10 /* a[m1...m2][n1...n2] と b[m1...m2][n1...n2] の和を求める。結果は cへ */
11 void matrix_sum(double **a, double **b, double **c, int m1, int m2, int n1, int n2);
12
13 int main(void)
14 {
15     double **a, **b, **c;
16     int i, j;
17
18     a = dmatrix(1, ROW, 1, COLUMN); /* 行列 a[1...ROW][1...COLUMN] */
19     b = dmatrix(1, ROW, 1, COLUMN); /* 行列 b[1...ROW][1...COLUMN] */
20     c = dmatrix(1, ROW, 1, COLUMN); /* 行列 c[1...ROW][1...COLUMN] */
21
22     /* 行列の定義 */
23     for(i = 1; i <= ROW; i++){
24         for(j = 1; j <= COLUMN; j++){
25             a[i][j] = 2.0 * (i + j); b[i][j] = 3.0 * (i + j);
26         }
27     }
28
29     /* 行列の和の計算 */
30     matrix_sum(a, b, c, 1, ROW, 1, COLUMN);
31
32     /* 結果の表示 */
33     printf("行列 A と行列 B の和は次の通りです\n");
34     for(i = 1; i <= ROW; i++){
35         for(j = 1; j <= COLUMN; j++){
36             printf("%f ", c[i][j]);
37         }
38         printf("\n");
39     }
40
41     /* 行列領域の解放 */
42     free_dmatrix(a, 1, ROW, 1, COLUMN);
43     free_dmatrix(b, 1, ROW, 1, COLUMN);
44     free_dmatrix(c, 1, ROW, 1, COLUMN);
45
46     return 0;
47 }
48
49 double **dmatrix(int nr1, int nr2, int n11, int n12)
50 {
51     int i, nrow, ncol;
52     double **a;
53
54     nrow = nr2 - nr1 + 1; /* 行の数 */
55     ncol = n12 - n11 + 1; /* 列の数 */
56
57     /* 行の確保 */
58     if((a = malloc(nrow * sizeof(double *))) == NULL){
59         printf("メモリが確保できません (行列 a)\n");
60         exit(1);
61     }
62     a = a - nr1; /* 行をずらす */
63
64     /* 列の確保 */
65     for(i = nr1; i <= nr2; i++) a[i] = malloc(ncol * sizeof(double));
66     for(i = nr1; i <= nr2; i++) a[i] = a[i] - n11; /* 列をずらす */
67
68     return (a);
69 }
70
71 void free_dmatrix(double **a, int nr1, int nr2, int n11, int n12)
72 {
73     int i;
74
75     /* メモリの解放 */
76     for(i = nr1; i <= nr2; i++) free((void *) (a[i] + n11));
77     free((void *) (a + nr1));
78 }
79
80 /* 行列の和 */
81 /* a[m1...m2][n1...n2] と b[m1...m2][n1...n2] の和を求める。結果は cへ */
82 void matrix_sum(double **a, double **b, double **c, int m1, int m2, int n1, int n2)
83 {
84     int i, j;
85     for(i = m1; i <= m2; i++){
86         for(j = n1; j <= n2; j++){
87             c[i][j] = a[i][j] + b[i][j];
88         }
89     }
90 }

```