```c
#include <stdio.h>
#include <stdlib.h>

#define N 4

/* 行列の入力 */
void input_matrix(double **a, char c, FILE *fin, FILE *fout);
/* ベクトルの入力 */
void input_vector(double *b, char c, FILE *fin, FILE *fout);

/* 行列の領域確保 */
double **dmatrix(int nr1, int nr2, int nl1, int nl2);
/* 行列の領域解放 */
void free_dmatrix(double **a, int nr1, int nr2, int nl, int nl2);
double *dvector(int i, int j); /* ベクトル領域の確保 */
void free_dvector(double *a, int i); /* 領域の解放 */

int main(void)
{
        FILE *fin, *fout;
        double **a, *b;

        /* 行列およびベクトルの領域確保 */
        a = dmatrix(1, N, 1, N);  /* 行列 a[1...N][1...N] */
        b = dvector(1, N);  /* b[1...N] */

        /* ファイルのオープン */
        if( (fin = fopen("input.dat", "r")) == NULL )
        {
                printf("ファイルが見つかりません : input.dat \n");
                exit(1);
        }

        if( (fout = fopen("output.dat", "w")) == NULL )
        {
                printf("ファイルが作成できません : output.dat \n");
                exit(1);
        }

        input_matrix(a, 'A', fin, fout);  /* 行列 A の入出力 */
        input_vector(b, 'b', fin, fout);  /* ベクトル b の入出力 */

        fclose(fin);  fclose(fout);

        /* 領域の解放 */
        free_dmatrix(a, 1, N, 1, N);
        free_dvector(b, 1);

        return 0;
}

/* a[1...N][1...N] の入力 */
void input_matrix(double **a, char c, FILE *fin, FILE *fout)
{
        int i, j;

        fprintf(fout, "行列%c は次の通りです\n", c);
        for(i = 1; i <= N; i++){
                for(j = 1; j <= N; j++){
                        fscanf(fin, "%lf", &a[i][j]);
                        fprintf(fout, "%5.2f\t", a[i][j]);
                }
                fprintf(fout, "\n");
        }
}

/* b[1...N]の入力 */
void input_vector(double *b, char c, FILE *fin, FILE *fout)
{
        int i;

        fprintf(fout, "ベクトル%c は次の通りです\n", c);
        for(i = 1; i <= N; i++){
                fscanf(fin, "%lf", &b[i]);
                fprintf(fout, "%5.2f\t", b[i]);
                fprintf(fout, "\n");
        }
}

double **dmatrix(int nr1, int nr2, int nl1, int nl2)
{
        int i, nrow, ncol;
        double **a;

        nrow = nr2 - nr1 + 1; /* 行の数 */
        ncol = nl2 - nl1 + 1; /* 列の数 */

        /* 行の確保 */
        if((a = malloc(nrow * sizeof(double *)) )== NULL){
                printf("メモリが確保できません (行列 a)\n");
                exit(1);
        }
        a = a - nr1; /* 行をずらす */

        /* 列の確保 */
        for(i = nr1; i <= nr2; i++) a[i] = malloc(ncol * sizeof(double));
        for(i = nr1; i <= nr2; i++) a[i] = a[i] - nl1;                  /* 列をずらす */

        return (a);
}

void free_dmatrix(double **a, int nr1, int nr2, int nl1, int nl2)
{
        int i;

        /* メモリの解放 */
        for(i = nr1; i <= nr2; i++) free((void *)(a[i] + nl1));
        free((void *)(a + nr1));
}

double *dvector(int i, int j)
{
        double *a;

        if((a = malloc( ((j - i + 1) * sizeof(double))) ) == NULL)
        {
                printf("メモリが確保できません (from dvector) \n");
                        exit(1);
        }

        return (a - i);
}

void free_dvector(double *a, int i)
{
        free( (void *)(a + i) ); /* (void *) 型へのキャストが必要 */
}
```