```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <unistd.h>
#include <sys/wait.h>

enum
{
    FILES = 0,
    FORMULES = 1,
};


// данные для примера 2
const int M = 2, N = 40; // пункт 2.2 приложение 2
const double pi = 3.1415926535;

void check_print(double **a, double *x, int n);
void init_matr(FILE *f, double **a, double *x, int n, int mode);
void mul_AX(double **a, double *x, int n);
void mul_AA(double **a, double **b, int n);
void cpy1(double *a, double *b, int n);
void cpy2(double **a, double **b, int n);
void transp(double **At, double **a, int n);
double norm(double *x, int n);
int relax(double **a, double *x, double *f, int n, double eps, double w);



int main(int argc, char **argv)
{
    FILE *fin = fopen(argv[1], "r");
    int n;
    fscanf(fin, "%d", &n);

    int mode;
    sscanf(argv[2], "%d", &mode);

    if (mode == 1)
        n = N;

    double **A, *x, *f;
    x = calloc(n, sizeof(double));
    f = calloc(n, sizeof(double));
    A = calloc(n, sizeof(double *));

    init_matr(fin, A, f, n, mode);

    double eps, w;
    sscanf(argv[3], "%lf", &eps);
    sscanf(argv[4], "%lf", &w);

    int steps = relax(A, x, f, n, eps, w);
    for(int i = 0; i < n; i++)
        printf("%.6lf ", x[i]);
    printf("  w = %.1lf: iterations = %d\n", w, steps);

    return 0;
}

int relax(double **a, double *x, double *f, int n, double eps, double w)
{
    int cnt = 0;
    double **At = calloc(n, sizeof(double *));
    for (int i = 0; i < n; i++)
        At[i] = calloc(n, sizeof(double));

    double *x_prev = calloc(n, sizeof(double));
    double *ft = calloc(n, sizeof(double));
```

```
70
71      transp(At, a, n);
72      mul_AA(At, a, n); // A = At * A
73      mul_AX(At, f, n); // b = At * b
74
75      do {
76          cnt++;
77          for(int i = 0; i < n; i++) {
78              x_prev[i] = x[i];
79          }
80          for(int i = 0; i < n; i++) {
81              double sum = 0;
82              for (int j = 0; j < i; j++) {
83                 sum += a[i][j] * x[j];
84              }
85
86              for (int j = i; j < n; j++) {
87                 sum += a[i][j] * x_prev[j];
88              }
89
90              x[i] = x_prev[i] + w * (f[i] - sum) / a[i][i];
91          }
92
93          for(int i = 0; i < n; i++) {
94              x_prev[i] = x[i];
95          }
96          mul_AX(a, x_prev, n); // x_prev = A * x
97          for(int i = 0; i < n; i++) {
98              x_prev[i] -= f[i];
99          }
100     } while (norm(x_prev, n) > eps);
101     free(At);
102     free(ft);
103     free(x_prev);
104     return cnt;
105 }
106
107 void transp(double **At, double **a, int n)
108 {
109     for(int i = 0; i < n; i++)
110         for(int j = 0; j < n; j++)
111             At[i][j] = a[j][i];
112
113 }
114
115 void cpy1(double *a, double *b, int n)
116 {
117     for(int i = 0; i < n; i++)
118         a[i] = b[i];
119 }
120
121 void cpy2(double **a, double **b, int n)
122 {
123     for(int i = 0; i < n; i++)
124         for(int j = 0; j < n; j++)
125             a[i][j] = b[i][j];
126 }
127
128
129 void mul_AX(double **a, double *x, int n)
130 {
131     double *res = calloc(n, sizeof(double));
132     for (int i = 0; i < n; i++){
133         for(int j = 0; j < n; j++){
134             res[i] += a[i][j] * x[j];
135         }
136     }
137     cpy1(x, res, n);
138     free(res);
```

```c
}

void mul_AA(double **a, double **b, int n)
{
    double **res = calloc(n, sizeof(double *));
    for (int i = 0; i < n; i++)
        res[i] = calloc(n, sizeof(double));

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            for (int k = 0; k < n; k++) {
                res[i][j] += a[i][k] * b[k][j];
            }
        }
    }

    cpy2(b, res, n);
    free(res);
}

double norm(double *x, int n)
{
    double res = 0. ;
    for (int i = 0; i < n; i++)
        res += x[i] * x[i];
    res = sqrt(res);
    return res;
}



void init_matr(FILE *fin, double **A, double *f, int n, int mode)
{
    double q = 1.001 - 2 * M * 0.001;

    for (int i = 0; i < n; i++)
        A[i] = calloc(n, sizeof(double));

    switch (mode) {
        case FILES:

            for (int i = 0; i < n; i++) {
                for (int j = 0; j < n; j++) {
                    fscanf(fin, "%lf", &A[i][j]);
                }
            }


            for (int i = 0; i < n; i++)
                fscanf(fin, "%lf", &f[i]);

            break;

        case FORMULES:
            for (int i = 0; i < n; i++) {
                for (int j = 0; j < n; j++) {
                    if (i == j) {
                        A[i][j] = pow(q-1, (double)(i+j));
                    } else {
                        A[i][j] = pow(q, (double)(i+j)) + 0.1 * (j - i);
                    }
                }
            }

            double x = pi/2; // случайный параметр для генерация вектора f
            for (int i = 0; i < n; i++)
                f[i] = fabs(x - N/10.) * i * sin(x);

            break;
```

```
208
209             default:
210                 return;
211         }
212
213         return;
214     }
215
216
217     void check_print(double **A, double *f, int n)
218     {
219         printf("A:\n");
220         for (int i = 0; i < n; i++) {
221             for (int j = 0; j < n; j++) {
222                 printf("%lf ", A[i][j]);
223             }
224             printf("\n");
225         }
226         printf("\nf: ");
227         for (int i = 0; i < n; i++)
228             printf("%lf ", f[i]);
229         printf("\n");
230     }
231
```