Московский Авиационный Институт (национальный исследовательский университет) Факультет «Прикладная математика и информатика» Кафедра 806

Курсовой проект по теме:

Разреженные матрицы

Выполнила студент группы М80-101Б-22

Соколова Виктория

Москва, 2023

Цель

Составить программу на языке Си с процедурами или функциями для обработки прямоугольных разреженных матриц с элементами комплексного типа.

Идея

Вычислить сумму двух матриц. Проверить, не является ли полученная матрица симметричной.

Входные файлы

test1:

2x3

1+0i 0 -1+3i

0 37+11i 0

2x3

-1+0i 0 1+-3i

0 -37+-11i 0

test2:

2x2

1+0i 0

0 0

2x2

0 0

0 1+0i

test3:

2x2

11+12i 0

0 0

2x2

0 1+0i

0 0

Тестирование

PS C:\code\test\kp\kp7> ./a

Use: program\_name input\_file

PS C:\code\test\kp\kp7> ./a test1

1.Print matrix 2. Task 3. Exit

1

Choose matrix: 1.A 2.B 3.C

2

Matrix size: 2x3

Internal representation:

LB

0 2 4

YE

-1.000000+0.000000i 1.000000+-3.000000i -37.000000+-11.000000i

Human readable:

-1.000000+0.000000i 0 1.000000+-3.000000i

0 -37.000000+-11.000000i 0

1.Print matrix 2. Task 3. Exit

sdfsf

No.

1.Print matrix 2. Task 3. Exit

1

Choose matrix: 1.A 2.B 3.C

dssf

No.

1.Print matrix 2. Task 3. Exit

2

C is symmetric matrix

1.Print matrix 2. Task 3. Exit

1

Choose matrix: 1.A 2.B 3.C

3

Matrix size: 2x3

Internal representation:

LB

YE

Human readable:

0 0 0

0 0 0

1.Print matrix 2. Task 3. Exit

1

Choose matrix: 1.A 2.B 3.C

1

Matrix size: 2x3

Internal representation:

LB

0 2 4

YE

1.000000+0.000000i -1.000000+3.000000i 37.000000+11.000000i

Human readable:

1.000000+0.000000i 0 -1.000000+3.000000i

0 37.000000+11.000000i 0

1.Print matrix 2. Task 3. Exit

3

PS C:\code\test\kp\kp7> ./a test2

1.Print matrix 2. Task 3. Exit

2

C is symmetric matrix

1.Print matrix 2. Task 3. Exit

1

Choose matrix: 1.A 2.B 3.C

3

Matrix size: 2x2

Internal representation:

LB

0 3

YE

1.000000+0.000000i 1.000000+0.000000i

Human readable:

1.000000+0.000000i 0

0 1.000000+0.000000i

1.Print matrix 2. Task 3. Exit

3

PS C:\code\test\kp\kp7> ./a test3

1.Print matrix 2. Task 3. Exit

2

C is not symmetric matrix

1.Print matrix 2. Task 3. Exit

1

Choose matrix: 1.A 2.B 3.C

3

Matrix size: 2x2

Internal representation:

LB

0 1

YE

11.000000+12.000000i 1.000000+0.000000i

Human readable:

11.000000+12.000000i 1.000000+0.000000i

0 0

1.Print matrix 2. Task 3. Exit

3

Код

#include <stdio.h>

#include <errno.h>

#include <stdlib.h>

struct complex {

    double re;

    double im;

};

struct matrix {

    int LB[100];

    struct complex YE[100];

    int M, N;

    int size;

};

struct matrix Read(FILE \*in) {

    struct matrix ans;

    ans.size = 0;

    struct complex c;

    fscanf(in, "%dx%d\n", &ans.M, &ans.N);

    for (int i = 0; i < ans.M; i++) {

        for (int j = 0; j < ans.N; j++) {

            if (fscanf(in, "%lf+%lfi", &c.re, &c.im) == 2) {

                ans.LB[ans.size] = i \* ans.N + j;

                ans.YE[ans.size] = c;

                ans.size++;

            }

        }

        fscanf(in, "\n");

    }

    ans.LB[ans.size] = -1;

    return ans;

}

void print\_matrix(struct matrix a) {

    if (!(a.M && a.N)){

        printf("Matrix doesn't exist\n");

        return;

    }

    printf("Matrix size: %dx%d\n", a.M, a.N);

    printf("Internal representation:\n");

    printf("LB\n");

    for (int i = 0; i < a.size; i++) {

        printf("%d\t", a.LB[i]);

    }

    printf("\nYE\n");

    for (int i = 0; i < a.size; i++) {

        printf("%lf+%lfi\t", a.YE[i].re, a.YE[i].im);

    }

    printf("\nHuman readable:\n");

    int k = 0;

    struct complex c;

    for (int i = 0; i < a.M; i++) {

        for (int j = 0; j < a.N; j++) {

            c.re = 0;

            c.im = 0;

            int h = i \* a.N + j;

            for (int ind = k; ind < a.size; ind++) {

                if (a.LB[ind] == h) {

                    c.re = a.YE[ind].re;

                    c.im = a.YE[ind].im;

                    k = ind;

                }

            }

            if (c.re != 0 || c.im != 0) {

                printf("%lf+%lfi\t", c.re, c.im);

            } else {

                printf("0\t");

            }

        }

        printf("\n");

    }

}

struct matrix sum(struct matrix A, struct matrix B) {

    if ((A.M != B.M) || (A.N != B.N)) {

        perror("Difficult matrix size");

        \_Exit(1);

    }

    struct matrix ans;

    ans.M = A.M;

    ans.N = A.N;

    ans.size = 0;

    struct complex ca, cb;

    int ka = 0, kb = 0;

    for (int i = 0; i < ans.M; i++) {

        for (int j = 0; j < ans.N; j++) {

            int h = i \* ans.N + j;

            ca.im = 0;

            ca.re = 0;

            cb.im = 0;

            cb.re = 0;

            for (int ind\_a = ka; ind\_a < A.size; ind\_a++) {

                if (A.LB[ind\_a] == h) {

                    ca = A.YE[ind\_a];

                    ka = ind\_a;

                    break;

                }

            }

            for (int ind\_b = kb; ind\_b < B.size; ind\_b++) {

                if (B.LB[ind\_b] == h) {

                    cb = B.YE[ind\_b];

                    kb = ind\_b;

                    break;

                }

            }

            struct complex answer;

            answer.re = ca.re + cb.re;

            answer.im = ca.im + cb.im;

            if (answer.re != 0 || answer.im != 0) {

                ans.LB[ans.size] = h;

                ans.YE[ans.size] = answer;

                ans.size++;

            }

        }

    }

    ans.LB[ans.size] = -1;

    return ans;

}

int sym(struct matrix A) {

    int ans = 1;

    for (int i = 0; i < A.size; i++) {

        int c = 0;

        int h = A.LB[i] / A.N + (A.LB[i] % A.N) \* A.N;

        for (int j = 0; j < A.size; j++) {

            if ((A.LB[j] == h) && (A.YE[i].re == A.YE[j].re) && (A.YE[i].im == A.YE[j].im)) {

                c = 1;

                break;

            }

        }

        if (!c) {

            ans = 0;

            break;

        }

    }

    return ans;

}

struct matrix task(struct matrix A, struct matrix B) {

    struct matrix C = sum(A, B);

    int c = sym(C);

    if (c) {

        printf("C is symmetric matrix\n");

    } else {

        printf("C is not symmetric matrix\n");

    }

    return C;

}

int main(int argc, char \*argv[]) {

    struct matrix A, B, C;

    int g = 1;

    int c;

    if (argc != 2) {

        printf("Use: program\_name input\_file\n");

        return 0;

    }

    FILE \*input = fopen(argv[1], "r");

    if (!input) {

        perror("Can't open file");

        return 1;

    }

    A = Read(input);

    B = Read(input);

    C.size = 0;

    int choose;

    while (g) {

        printf("1.Print matrix  2. Task  3. Exit\n");

        if (scanf("%d", &c) != 1){

            printf("No.\n");

            fflush(stdin);

            continue;

        }else{

            if (c == 1){

                printf("Choose matrix: 1.A   2.B    3.C\n");

                if (scanf("%d", &choose) != 1){

                    printf("No.\n");

                    fflush(stdin);

                    continue;

                }else{

                    if(choose == 1){

                        print\_matrix(A);

                    } else if(choose == 2){

                        print\_matrix(B);

                    } else if(choose == 3){

                        print\_matrix(C);

                    } else{

                        printf("Wrong answer\n");

                    }

                }

            } else if (c == 2){

                C =  task(A, B);

            }else if (c == 3){

                break;

            }else {

                printf("Wrong answer\n");

            }

        }

    }

}