**Отчет по курсовому проекту №7** по курсу фундаментальная информатика

Студентка группы М8О-113Б-22, № по списку 1, Астахова Анастасия Сергеевна

Контакты astakhovaanastasia0201@gmail.com

Работа выполнена: « » \_\_\_\_\_\_\_\_ 2023 г.

Преподаватель: доцент каф. 806 Никулин Сергей Петрович

Входной контроль знаний с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет сдан « » \_\_\_\_\_\_\_\_\_2023 г., итоговая оценка \_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* 1. **Тема:** Разреженные матрицы

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Цель работы:** Составить и отладить программу на языке Си с процедурами и/или функциями для обработки прямоугольных разреженных матриц элементами вещественного типа.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Задание (схема размещения матрицы№1, преобразование №1, физическое представление №2):** схема размещения матрицы - цепочка ненулевых элементов в векторе А со строчным индексированием, преобразование - определить максимальный по модулю элемент матрицы и разделить на него все элементы строки, в которой он находится. Если таких элементов несколько, обработать каждую строку, содержащую такой элемент.физическое представление – отображение на динамические структуры \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. **Оборудование** (лабораторное):

ЭВМ \_\_\_\_\_\_\_\_\_, процессор \_\_\_\_ \_\_, имя узла сети \_\_\_\_\_\_ с ОП \_\_\_\_\_\_\_\_ Мб, НМД \_\_\_\_\_\_\_ Мб. Терминал \_\_\_\_\_ адрес \_\_\_\_\_\_\_\_\_\_. Принтер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Другие устройства \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Оборудование ПЭВМ студента, если использовалось:*

Ноутбук: Процессор Intel Core i5-10210U с ОП 16 ГБ, SSD 512 ГБ. Другие устройства: мышь Bloody R80

*Компьютер: Процессор Ryzen 5 3600 с ОП 16 ГБ, SSD 512 ГБ. Другие устройства: мышь* Razer BASILISK X HYPERSPEED, [*Logitech G910 (920-008019) Orion Spectrum RGB Mechanical Gaming*](https://www.komus.ru/katalog/tekhnika/kompyuternaya-tekhnika/klaviatury-i-myshi/klaviatury-provodnye/klaviatura-logitech-g910-920-008019-orion-spectrum-rgb-mechanical-gaming/p/659916/?from=n-13776), монитор [*LG UltraGear 27GP750*](https://www.lg.com/ru/monitors/lg-27gp750-b)

1. **Программное обеспечение (лабораторное):**

Операционная система семейства \_\_\_Unix\_\_\_\_, наименование \_\_\_\_\_Ubuntu\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_4.15.0\_\_\_\_\_\_

интерпретатор команд \_\_\_\_bash\_\_\_\_\_\_\_\_ версия \_\_\_4.4.20\_\_\_\_

Система программирования \_\_С\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_\_\_\_\_\_последняя\_\_\_\_\_\_\_\_\_

Редактор текстов \_\_\_gedit\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_25.2.2 \_\_\_\_\_\_\_\_\_

Утилиты операционной системы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Прикладные системы и программы \_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Местонахождение и имена файлов программ и данных\_\_stud/208104 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**5.** *Программное обеспечение ПЭВМ студента, если использовалось****:***

Ноутбук: Операционная система семейства Ubuntu, наименование версия Ubuntu 20.04.3 LTS,

*интерпретатор команд bash версия 5.0.17. Система программирования C. Редактор текстов gedit версия 28.2. Утилиты: стандартные утилиты UNIX. Программное обеспечение: стандартное программное обеспечение системы UNIX.*

**6. Идея, метод, алгоритм** решения задачи(в формах:словесной,псевдокода,графической[блок-схема,диаграмма,рисунок,таблица] или формальные спецификации с пред- и постусловиями)

Будем хранить в массиве два динамических линейных списка, каждый элемент которого будет следующую информацию: индекс (голова начинается с индекса 1), значение и указатель на следующий элемент. Это структура Item.

Для Item создадим следующие функции:

- Item\* init(int number) – Возвращает инициализированный список с нулевыми значениями number элементов.

- Item\* getLast(Item\* head) – Возвращает последний элемент списка head

- Item\* Add(Item\* head, double val) – Добавляет в конец списка (даже равный NULL) элемент со значением val и возвращает укзаатель на голову.

- Item\* getByIndex(Item\* head, int ind) – Возвращает элемент списка head с индексом ind (напоминаю, 1-индексация)

- int getSize(Item\* head) – возвращает кол-во элементов списка

- void printItems(Item\* head) – пишет все элементы списка через пробел

В матрице хранятся два соответствующих списка A и M, размерность матрицы, а также значение MX, отвечающее за максимум по модулю. Calcline – вспомогательная функция для task, а это функция деления строк на найденный максимум при чтении матрицы. Все остальные функции интуитивно понятны.

1. **Сценарий выполнения работы** [план работы,первоначальный текст программы в черновике(можно на отдельном листе)итесты либо соображения по тестированию

#include <stdio.h>

#include <errno.h>

#include <stdlib.h>

#include <math.h>

//На основе списка

typedef struct {

int index;

struct Item\* next;

double value;

} Item;

Item\* init(int number) {

Item\* head = malloc(sizeof(Item));

head->next = NULL;

head->value = 0;

head->index = 1;

Item\* cur = head;

for (int i = 1; i < number; i++) {

Item\* tmp = malloc(sizeof(Item));

tmp->next = NULL;

tmp->value = 0;

tmp->index = i + 1;

cur->next = tmp;

cur = cur->next;

}

return head;

}

Item\* getLast(Item\* head) {

if (head == NULL) return NULL;

Item\* g = head;

while (g->next != NULL) {

g = g->next;

}

return g;

}

Item\* Add(Item\* head, double val) {

if (head == NULL) {

Item\* tmp = malloc(sizeof(Item));

tmp->next = NULL;

tmp->value = val;

tmp->index = 1;

return tmp;

}

Item\* last = getLast(head);

Item\* tmp = malloc(sizeof(Item));

tmp->next = NULL;

tmp->value = val;

tmp->index = last->index + 1;

last->next = tmp;

return head;

}

Item\* getByIndex(Item\* head, int ind) {

if (head == NULL) return NULL;

Item\* g = head;

while (g->index != ind && g->next != NULL)

g = g->next;

return g;

}

int getSize(Item\* head) {

Item\* last = getLast(head);

if (last == NULL) return 0;

return last->index;

}

void printItems(Item\* head) {

Item\* g = head;

while (g != NULL) {

printf("%lf ", g->value);

g = g->next;

}

printf("\n");

}

typedef struct {

Item\* M;

Item\* A;

int m, n;

double MX;

} Matrix;

Matrix\* initi(int Mm, int Nn) {

Matrix\* ans = malloc(sizeof(Matrix));

ans->m = Mm;

ans->n = Nn;

ans->M = init(Mm);

ans->A = NULL;

return ans;

}

//1-индексация хранения!!!

Matrix\* readMatrix(FILE\* in) {

double curVal = 0;

int m, n;

int curRow = 0;

fscanf(in, "%d %d\n", &m, &n);

Matrix\* ans = initi(m, n);

for (int i = 0; i < m; i++) {

for (int j = 0; j < n; j++) {

if (fscanf(in, "%lf", &curVal) == 1) {

if (curVal == 0)

continue;

int realI = i + 1;

int realJ = j + 1;

if (curRow == 0) {

curRow = realI;

Item\* Mind = getByIndex(ans->M, realI);

Mind->value = 1;

}

//Переход на новую строку

else if (curRow != realI) {

//Записали - как символ окончания строки

ans->A = Add(ans->A, 0);

curRow = realI;

Item\* Mind = getByIndex(ans->M, realI);

//+1? поскольку не хотим указывать на 0

Mind->value = getSize(ans->A) + 1;

}

ans->A = Add(ans->A, realJ);

ans->MX = (abs(curVal)) > (ans->MX) ? (abs(curVal)) : (ans->MX);

ans->A = Add(ans->A, curVal);

}

}

fscanf(in, "\n");

}

//Последний 0 нужно записать в Item

ans->A = Add(ans->A, 0);

return ans;

}

double EPS = 1e-6;

//1-индексация!!!

void printMatrix(Matrix\* A) {

printf("Human representation:\n");

for (int i = 1; i <= A->m; i++) {

Item\* mmm = getByIndex(A->M, i);

if (mmm->value == 0) {

for (int j = 1; j <= A->n; j++) {

printf("%d\t", 0);

}

}

else {

Item\* ColumnsAndValues = getByIndex(A->A, mmm->value);

for (int j = 1; j <= A->n; j++) {

if (ColumnsAndValues->value != 0 && j == ColumnsAndValues->value) {

ColumnsAndValues = ColumnsAndValues->next;

printf("%.2lf\t", ColumnsAndValues->value);

ColumnsAndValues = ColumnsAndValues->next;

}

else {

printf("%d\t", 0);

}

}

}

printf("\n");

}

printf("Internal Represenation:\n");

printf("M:\n");

printItems(A->M);

printf("A:\n");

printItems(A->A);

}

Item\* calcLine(Item\* headLine, double val) {

while (headLine->value != 0) {

Item\* nxt = headLine->next;

nxt->value = nxt->value / val;

headLine = nxt->next;

}

return headLine;

}

Matrix\* task(Matrix \*A) {

double modMax = A->MX;

Item\* headLine = A->A;

Item\* curIndex = headLine;

while (curIndex != NULL) {

while (curIndex->value != 0) {

Item\* nxt = curIndex->next;

double val = nxt->value;

if (abs(val) == modMax) {

headLine = calcLine(headLine, val);

//останавливается на 0

}

curIndex = nxt->next;

}

headLine = headLine->next;

curIndex = curIndex->next;

}

return A;

}

int main() {

Matrix\* A;

char path[256];

printf("Enter file input: ");

scanf("%s", path);

FILE\* input = fopen(path, "r");

if (input == NULL) {

printf("Error: file not found\n");

return 0;

}

A = readMatrix(input);

fclose(input);

int g = 1;

while (g) {

printf("1) Print A\n");

printf("2) Task\n");

printf("3) Exit\n");

int command;

scanf("%d", &command);

switch (command) {

case 1: {

printMatrix(A);

break;

}

case 2: {

A = task(A);

break;

}

case 3: {

g = 0;

break;

}

default: {

printf("Error: unknown command");

break;

}

}

}

return 0;

}

Тесты:

first — максимальный элемент по модулю — это 5 и -5. Поэтому соотв. Строки будут разделены на эти числа

5 5

0 0 0 0 5

0 0 -5 3 2

0 0 0 0 1

3 0 0 0 4

second — максимальный элемент 10.

1 4

0 0 10 0

third — максимальный элемент 2. Разделены будут 1, 2 и 4 строки.

3 2

0 2

1 0

2 1

*Пункты 1-7 отчета составляются сторого до начала лабораторной работы.*

*Допущен к выполнению работы.* **Подпись преподавателя****\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. **Распечатка протокола** (подклеить листинг окончательного варианта программы с тестовыми примерами,подписанныйпреподавателем).

**kokodrillo@kokodrillo-VirtualBox:~/work/ВТОРОЙ\_СЕМ/for\_lab\_7$ ls**

**first main.c second third**

**kokodrillo@kokodrillo-VirtualBox:~/work/ВТОРОЙ\_СЕМ/for\_lab\_7$ cat main.c**

**#include <stdio.h>**

**#include <errno.h>**

**#include <stdlib.h>**

**#include <math.h>**

**//На основе списка**

**typedef struct {**

**int index;**

**struct Item\* next;**

**double value;**

**} Item;**

**Item\* init(int number) {**

**Item\* head = malloc(sizeof(Item));**

**head->next = NULL;**

**head->value = 0;**

**head->index = 1;**

**Item\* cur = head;**

**for (int i = 1; i < number; i++) {**

**Item\* tmp = malloc(sizeof(Item));**

**tmp->next = NULL;**

**tmp->value = 0;**

**tmp->index = i + 1;**

**cur->next = tmp;**

**cur = cur->next;**

**}**

**return head;**

**}**

**Item\* getLast(Item\* head) {**

**if (head == NULL) return NULL;**

**Item\* g = head;**

**while (g->next != NULL) {**

**g = g->next;**

**}**

**return g;**

**}**

**Item\* Add(Item\* head, double val) {**

**if (head == NULL) {**

**Item\* tmp = malloc(sizeof(Item));**

**tmp->next = NULL;**

**tmp->value = val;**

**tmp->index = 1;**

**return tmp;**

**}**

**Item\* last = getLast(head);**

**Item\* tmp = malloc(sizeof(Item));**

**tmp->next = NULL;**

**tmp->value = val;**

**tmp->index = last->index + 1;**

**last->next = tmp;**

**return head;**

**}**

**Item\* getByIndex(Item\* head, int ind) {**

**if (head == NULL) return NULL;**

**Item\* g = head;**

**while (g->index != ind && g->next != NULL)**

**g = g->next;**

**return g;**

**}**

**int getSize(Item\* head) {**

**Item\* last = getLast(head);**

**if (last == NULL) return 0;**

**return last->index;**

**}**

**void printItems(Item\* head) {**

**Item\* g = head;**

**while (g != NULL) {**

**printf("%lf ", g->value);**

**g = g->next;**

**}**

**printf("\n");**

**}**

**typedef struct {**

**Item\* M;**

**Item\* A;**

**int m, n;**

**double MX;**

**} Matrix;**

**Matrix\* initi(int Mm, int Nn) {**

**Matrix\* ans = malloc(sizeof(Matrix));**

**ans->m = Mm;**

**ans->n = Nn;**

**ans->M = init(Mm);**

**ans->A = NULL;**

**return ans;**

**}**

**//1-индексация хранения!!!**

**Matrix\* readMatrix(FILE\* in) {**

**double curVal = 0;**

**int m, n;**

**int curRow = 0;**

**fscanf(in, "%d %d\n", &m, &n);**

**Matrix\* ans = initi(m, n);**

**for (int i = 0; i < m; i++) {**

**for (int j = 0; j < n; j++) {**

**if (fscanf(in, "%lf", &curVal) == 1) {**

**if (curVal == 0)**

**continue;**

**int realI = i + 1;**

**int realJ = j + 1;**

**if (curRow == 0) {**

**curRow = realI;**

**Item\* Mind = getByIndex(ans->M, realI);**

**Mind->value = 1;**

**}**

**//Переход на новую строку**

**else if (curRow != realI) {**

**//Записали - как символ окончания строки**

**ans->A = Add(ans->A, 0);**

**curRow = realI;**

**Item\* Mind = getByIndex(ans->M, realI);**

**//+1? поскольку не хотим указывать на 0**

**Mind->value = getSize(ans->A) + 1;**

**}**

**ans->A = Add(ans->A, realJ);**

**ans->MX = (abs(curVal)) > (ans->MX) ? (abs(curVal)) : (ans->MX);**

**ans->A = Add(ans->A, curVal);**

**}**

**}**

**fscanf(in, "\n");**

**}**

**//Последний 0 нужно записать в Item**

**ans->A = Add(ans->A, 0);**

**return ans;**

**}**

**double EPS = 1e-6;**

**//1-индексация!!!**

**void printMatrix(Matrix\* A) {**

**printf("Human representation:\n");**

**for (int i = 1; i <= A->m; i++) {**

**Item\* mmm = getByIndex(A->M, i);**

**if (mmm->value == 0) {**

**for (int j = 1; j <= A->n; j++) {**

**printf("%d\t", 0);**

**}**

**}**

**else {**

**Item\* ColumnsAndValues = getByIndex(A->A, mmm->value);**

**for (int j = 1; j <= A->n; j++) {**

**if (ColumnsAndValues->value != 0 && j == ColumnsAndValues->value) {**

**ColumnsAndValues = ColumnsAndValues->next;**

**printf("%.2lf\t", ColumnsAndValues->value);**

**ColumnsAndValues = ColumnsAndValues->next;**

**}**

**else {**

**printf("%d\t", 0);**

**}**

**}**

**}**

**printf("\n");**

**}**

**printf("Internal Represenation:\n");**

**printf("M:\n");**

**printItems(A->M);**

**printf("A:\n");**

**printItems(A->A);**

**}**

**Item\* calcLine(Item\* headLine, double val) {**

**while (headLine->value != 0) {**

**Item\* nxt = headLine->next;**

**nxt->value = nxt->value / val;**

**headLine = nxt->next;**

**}**

**return headLine;**

**}**

**Matrix\* task(Matrix \*A) {**

**double modMax = A->MX;**

**Item\* headLine = A->A;**

**Item\* curIndex = headLine;**

**while (curIndex != NULL) {**

**while (curIndex->value != 0) {**

**Item\* nxt = curIndex->next;**

**double val = nxt->value;**

**if (abs(val) == modMax) {**

**headLine = calcLine(headLine, val);**

**//останавливается на 0**

**}**

**curIndex = nxt->next;**

**}**

**headLine = headLine->next;**

**curIndex = curIndex->next;**

**}**

**return A;**

**}**

**int main() {**

**Matrix\* A;**

**char path[256];**

**printf("Enter file input: ");**

**scanf("%s", path);**

**FILE\* input = fopen(path, "r");**

**if (input == NULL) {**

**printf("Error: file not found\n");**

**return 0;**

**}**

**A = readMatrix(input);**

**fclose(input);**

**int g = 1;**

**while (g) {**

**printf("1) Print A\n");**

**printf("2) Task\n");**

**printf("3) Exit\n");**

**int command;**

**scanf("%d", &command);**

**switch (command) {**

**case 1: {**

**printMatrix(A);**

**break;**

**}**

**case 2: {**

**A = task(A);**

**break;**

**}**

**case 3: {**

**g = 0;**

**break;**

**}**

**default: {**

**printf("Error: unknown command");**

**break;**

**}**

**}**

**}**

**return 0;**

**}**

**kokodrillo@kokodrillo-VirtualBox:~/work/ВТОРОЙ\_СЕМ/for\_lab\_7$ cat first**

**5 5**

**0 0 0 0 5**

**0 0 -5 3 2**

**0 0 0 0 1**

**3 0 0 0 4**

**kokodrillo@kokodrillo-VirtualBox:~/work/ВТОРОЙ\_СЕМ/for\_lab\_7$ cat second**

**1 4**

**0 0 10 0**

**kokodrillo@kokodrillo-VirtualBox:~/work/ВТОРОЙ\_СЕМ/for\_lab\_7$ cat third**

**3 2**

**0 2**

**1 0**

**2 1**

**kokodrillo@kokodrillo-VirtualBox:~/work/ВТОРОЙ\_СЕМ/for\_lab\_7$ gcc main.c**

**kokodrillo@kokodrillo-VirtualBox:~/work/ВТОРОЙ\_СЕМ/for\_lab\_7$ ./a.out**

**Enter file input: first**

**1) Print A**

**2) Task**

**3) Exit**

**1**

**Human representation:**

**0 0 0 0 5.00**

**0 0 -5.00 3.00 2.00**

**0 0 0 0 1.00**

**3.00 0 0 0 4.00**

**0 0 0 0 0**

**Internal Represenation:**

**M:**

**1.000000 4.000000 11.000000 14.000000 0.000000**

**A:**

**5.000000 5.000000 0.000000 3.000000 -5.000000 4.000000 3.000000 5.000000 2.000000 0.000000 5.000000 1.000000 0.000000 1.000000 3.000000 5.000000 4.000000 0.000000**

**1) Print A**

**2) Task**

**3) Exit**

**2**

**1) Print A**

**2) Task**

**3) Exit**

**1**

**Human representation:**

**0 0 0 0 1.00**

**0 0 1.00 -0.60 -0.40**

**0 0 0 0 1.00**

**3.00 0 0 0 4.00**

**0 0 0 0 0**

**Internal Represenation:**

**M:**

**1.000000 4.000000 11.000000 14.000000 0.000000**

**A:**

**5.000000 1.000000 0.000000 3.000000 1.000000 4.000000 -0.600000 5.000000 -0.400000 0.000000 5.000000 1.000000 0.000000 1.000000 3.000000 5.000000 4.000000 0.000000**

**1) Print A**

**2) Task**

**3) Exit**

**3**

**kokodrillo@kokodrillo-VirtualBox:~/work/ВТОРОЙ\_СЕМ/for\_lab\_7$ ./a.out**

**Enter file input: second**

**1) Print A**

**2) Task**

**3) Exit**

**1**

**Human representation:**

**0 0 10.00 0**

**Internal Represenation:**

**M:**

**1.000000**

**A:**

**3.000000 10.000000 0.000000**

**1) Print A**

**2) Task**

**3) Exit**

**2**

**1) Print A**

**2) Task**

**3) Exit**

**1**

**Human representation:**

**0 0 1.00 0**

**Internal Represenation:**

**M:**

**1.000000**

**A:**

**3.000000 1.000000 0.000000**

**1) Print A**

**2) Task**

**3) Exit**

**3**

**kokodrillo@kokodrillo-VirtualBox:~/work/ВТОРОЙ\_СЕМ/for\_lab\_7$ ./a.out**

**Enter file input: third**

**1) Print A**

**2) Task**

**3) Exit**

**1**

**Human representation:**

**0 2.00**

**1.00 0**

**2.00 1.00**

**Internal Represenation:**

**M:**

**1.000000 4.000000 7.000000**

**A:**

**2.000000 2.000000 0.000000 1.000000 1.000000 0.000000 1.000000 2.000000 2.000000 1.000000 0.000000**

**1) Print A**

**2) Task**

**3) Exit**

**2**

**1) Print A**

**2) Task**

**3) Exit**

**1**

**Human representation:**

**0 1.00**

**1.00 0**

**1.00 0.50**

**Internal Represenation:**

**M:**

**1.000000 4.000000 7.000000**

**A:**

**2.000000 1.000000 0.000000 1.000000 1.000000 0.000000 1.000000 1.000000 2.000000 0.500000 0.000000**

**1) Print A**

**2) Task**

**3) Exit**

**3**

1. **Дневник отладки** должен содержать дату и время сеансов отладки, и основные события(ошибки в сценарии и программе,нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  | или |  |  |  |  |  |
|  | дом. |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. **Замечания автора** по существу работы:

Замечания отсутствуют.

1. **Выводы**

Я научилась работать с разреженными матрицами.

Недочёты при выполнении задания могут быть устранены следующим образом:

Недочеты отсутствуют.

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_