|  |  |
| --- | --- |
| итдля прик эмбл | |
| МИНОБРНАУКИ РОССИИ | |
| Федеральное государственное бюджетное образовательное учреждениевысшего образования«Московский государственный университет информационных технологий, радиотехники и электроники»МИРЭА | |
| Институт информационных технологий (ИТ) | |
| Базовая кафедра № 239 «АСОУ» МИРЭА при ФГУП НИИ «Восход» | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО *УЧЕБНОЙ* ПРАКТИКЕ** | |
| **Тема практики «**\_Технологии программирования абстрактных типов данных в с++ и в Matlab \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**»**  *(наименование темы)* | |
| приказ университета о направлении на практику от «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 201\_\_ г. № \_\_\_\_\_\_\_ | |
| Студент группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(учебная группа)* | *Фамилия И.О.* |
| Руководитель практики  *должность, звание, ученая степень* | *Фамилия И.О.* |

|  |  |  |
| --- | --- | --- |
| «Отчет представлен к рассмотрению» | «\_\_»\_\_\_\_\_\_\_\_\_\_201\_\_\_ г. |  |
|  |  |  |
| «Отчет утвержден. Допущен к защите» | «\_\_»\_\_\_\_\_\_\_\_\_\_201\_\_\_ г. |  |

Москва 2015

**УТВЕРЖДАЮ**

Заведующий базовой кафедрой «АСОУ» (№ 239) МИРЭА при ФГУП НИИ «Восход»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.В. Волков

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2015 г.

**ЗАДАНИЕ ПО УЧЕБНОЙ ПРАКТИКЕ**

**студента *Панасенко А.В***

**группы *ИВБ-3-14***

Руководитель практики от кафедры – *ст. преподаватель Рачковская Евгения Федоровна*

Руководитель практики от предприятия – *Начальник отдела Научно-учебного департамента ФГУП НИИ «Восход» Рачковская Евгения Федоровна ФИО.*

1. **Общая часть задания**
   1. Пройти инструктаж по правилам пожарной безопасности, по охране труда на рабочем месте и по технике безопасности и производственной санитарии для пользователей ПЭВМ.
   2. Оформить задание на практику.
   3. Принять участие в собрании по организации учебной практики.
   4. Принять участие в мероприятии по презентации результатов практики.
2. **Индивидуальное задание**

|  |  |  |
| --- | --- | --- |
| «Методы сортировки и поиска, технологии разработки АТД в виде классов с++, стандартные средства обработки матриц и визуализации данных в MatLab» | | |
| № | Содержание основных частей задания | Планируемый результат |
|  | Изучение предметной области | Изучение методов сортировки и поиска |
|  | Разработка АТД | Описание АТД |
|  | Изучение Matlab | База знаний по Matlab |
|  | Разработка алгоритмов реализации операций АТД | Алгоритмы сортировки:   * методом Выбора * методом Шелла * методом Слияния   Алгоритмы поиска:   * Поиск прямой * Поиск Фиббоначи   Алгоритмы обработки матриц: |
|  | Реализация АТД в виде классов с++ | Программный код |
|  | Реализация операций над векторами и матрицами средствами Matlab в соответствии с заданием | Программный код |
|  | Визуализация данных средствами Matlab в соответствии с заданием. | Программный код, реферат «Графика векторов» |
|  | Подготовка отчета по практике | Отчет, презентация |
|  | Подготовка к зачету по практике | Доклад |
|  | Зачет по практике | Зачет |

**КАЛЕНДАРНО-ТЕМАТИЧЕСКИЙ ПЛАН ПРАКТИКИ**

Студент: *Панасенко Андрей Викторович*

Место проведения практики: ФГУП НИИ «Восход»

Индивидуальное задание на практику: «Методы сортировки и поиска, технологии разработки АТД в виде классов с++, стандартные средства обработки матриц и визуализации данных в MatLab»

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Содержание работ** | **Сроки выполнения (дни практики)** | **Примечания** |
|
|  | Формирование и утверждение индивидуального задания на практику. | 1 |  |
|  | Установочная лекция | 1 |  |
|  | Изучение предметной области | 2 |  |
|  | Разработка АТД:   * «my\_vector»; * «matrix»;   в соответствии с заданием | 3 |  |
|  | Изучение Matlab:   * структура программного комплекса; * язык программирования; * матричные операции; * средства визуализации данных. | 4-6 |  |
|  | Разработка алгоритмов реализации операций АТД | 7 |  |
|  | Реализация АТД в виде классов с++ | 8 |  |
|  | Реализация операций над векторами и матрицами средствами Matlab в соответствии с заданием | 9 |  |
|  | Визуализация данных средствами Matlab в соответствии с заданием. | 10 |  |
|  | Подготовка отчета по практике | 11-12 |  |
|  | Подготовка к зачету по практике | 13 |  |
|  | Зачет по практике | 14 |  |

|  |  |  |
| --- | --- | --- |
| «Задание выдал» | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_ 201\_\_г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* |
| «Задание выдал» | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_ 201\_\_г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* |
| «Задание получил» | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_ 201\_\_г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* |

**Прохождение инструктажей**

|  |  |  |
| --- | --- | --- |
| Мероприятие | ФИО ответственного, подпись и дата | ФИО, подпись студента и дата |
| Проведение инструктажа по правилам пожарной безопасности |  |  |
| Проведение инструктажа по охране труда на рабочем месте |  |  |
| Проведение инструктажа по технике безопасности и производственной санитарии для пользователей ПЭВМ |  |  |

**Отметки о прохождении практики**

|  |  |  |
| --- | --- | --- |
| Дни практики | Отметка о посещении | Краткая характеристика работы |
| 1 |  | Формирование и утверждение индивидуального задания на практику. |
| 1 |  | Установочная лекция |
| 2 |  | Изучение предметной области |
| 3 |  | Разработка АТД |
| 4-6 |  | Изучение Matlab |
| 7 |  | Разработка алгоритмов реализации операций АТД |
| 8 |  | Реализация АТД в виде классов с++ |
| 9 |  | Реализация операций над векторами и матрицами средствами Matlab в соответствии с заданием |
| 10 |  | Визуализация данных средствами Matlab в соответствии с заданием. |
| 11-12 |  | Подготовка отчета по практике |
| 13 |  | Подготовка к зачету по практике |
| 14 |  | Зачет по практике |

**ОТЗЫВ – ХАРАКТЕРИСТИКА О ПРОХОЖДЕНИИ ПРАКТИКИ**

Студент базовой кафедры № 239 «АСОУ» МИРЭА при ФГУП НИИ «Восход» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(Фамилия Имя Отчество)*

обучающийся на \_\_1\_\_ курсе Института \_\_\_информационных технологий\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

в группе \_\_\_\_\_\_\_\_\_ИБВ-3-14\_\_\_\_\_\_\_\_\_\_\_\_\_\_ по направлению подготовки (специальности) \_\_\_\_\_09.03.01\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ проходил \_\_\_\_\_учебную\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ практику

*вид практики (производственная, преддипломная и т.д.)*

с «\_01\_» \_\_\_\_\_июля\_\_\_\_\_ 2015 г. по «\_14\_»\_\_\_\_июля\_\_\_\_ 2015 г.в \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ФГУП НИИ «Восход»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

*(наименование предприятия, структурного подразделения Университета)*

Руководитель практики от предприятия: начальник отдела научно-учебного департамента\_\_

ФГУП НИИ «Восход»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Практика проводилась в отделе (лаборатории): Научно-учебном департаменте\_\_\_\_\_\_\_\_\_\_\_.

Во время прохождения практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(Фамилия И.О.)*

исполнял следующие обязанности:\_\_программиста\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

приобрел следующие профессиональные компетенции:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_1. Навыки разработки АТД и реализации их в виде классов С++\_и в среде Matlab.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_2. Навыки программирования в среде Matlab.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

проявил себя как: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ и представил отчет по практике.

Задание по практике выполнено в полном объеме, студент заслуживает оценки «\_\_\_\_\_\_\_\_\_\_\_\_\_\_».

Руководитель практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_Е.Ф. Рачковская\_\_\_\_/\_\_\_

*(подпись) (расшифровка подписи)*

**содержание**

[Введение. Цели и задачи проведения практики 6](#_Toc424556650)

[Основная часть 7](#_Toc424556651)

[Задание 1. АТД Вектор 7](#_Toc424556652)

[Цель работы 7](#_Toc424556653)

[Задание 7](#_Toc424556654)

[Задание 2. АТД Матрица 22](#_Toc424556655)

[Цель работы 22](#_Toc424556656)

[Задание 22](#_Toc424556657)

[Задание 3. Графика в Matlab 36](#_Toc424556658)

[Цель работы 36](#_Toc424556659)

[Задание 36](#_Toc424556660)

Заключение 39

Список используемой литературы 40

# Введение. Цели и задачи проведения практики

Учебная практика (УП) предполагает подготовку студентов в области современных информационных технологий, методов и программных средств, используемых на прикладном уровне.

1. Основные цели практики:

* закрепление и углубление теоретической подготовки обучающегося по программированию;
* изучение основ математического пакета MatLab;
* приобретение навыков самостоятельной и научно-исследовательской работы;
* приобретение опыта применения современной вычислительной техники и современных инструментальных средств для решения практических задач;
* приобретение практических навыков и компетенций в сфере профессиональной деятельности.

1. Задачи проведения практики:

* знакомство с основами будущей профессиональной деятельности;
* приобретение студентами практических навыков, знаний и умений самостоятельной разработки абстрактных типов данных;
* освоение технологии применения абстрактных типов данных в практике программирования на с++;
* приобретение студентами практических навыков обработки и визуализации данных в среде программирования MatLab.
* выработка навыков разработки презентаций выполненной работы, навыков составления научно-технических отчетов, навыков составления обзоров научной литературы и электронных информационно-образовательных ресурсов для профессиональной деятельности.

# Основная часть

## Задание 1. АТД Вектор

### Цель работы

1. Приобрести навыки проектирования абстрактных типов данных (АТД).
2. Приобрести навыки разработки класса в с++.
3. Изучить и реализовать методы сортировки одномерных массивов и методы поиска элемента в массиве. Провести экспериментальное исследование эффективности методов сортировки и поиска.
4. Приобрести навыки программирования на языке программирования Matlab.

### Задание

1. На базе шаблона класса **vector** разработать, реализовать и провести тестовые испытания) ***АТД* «my\_vector»** путем добавления следующих методов (алгоритмов):

* вывод вектора на экран;
* заполнение вектора с помощью датчика случайных чисел;
* sort1, sort2, , sort3, реализующие операции сортировки массива указанными в варианте задания методами;
* find 1, find2, реализующие операции поиска элемента в упорядоченном массиве указанным в варианте задания методами

Таблица 1 – Вариант заданий по методам сортировки и поиска

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № вар | Метод сортировки 1 | Метод сортировки 2 | Метод сортировки 3 | Метод поиска 1 | Метод поиска 2 |
| Чт | Метод выбора | Сортировка Шелла | Сортировка слиянием | Прямой | Фибоначчи |

Для тестирования эффективности алгоритмов интерфейс разрабатываемого АТД должен включать следующие дополнительные операции:

* + опрос числа выполненных сравнений;
  + опрос числа выполненных обменов.

1. Реализовать АТД **«my\_vector»** в виде класса C++.
2. Разработать наборы тестов для тестирования операций над массивами в разрабатываемых приложениях.
3. Выполнить отладку и тестирование отдельных операций АТД с помощью меню операций.
4. Реализовать средствами Matlab набор операций над массивом, представленными в АТД **«my\_vector»** и провести их тестирование, разработав соответствующее приложение.

***Формат АТД***

**АТД MyVector**

Автоматический тип данных/шаблон (Type)

**ДАННЫЕ:**

Вектор типа Type.

Шаблон, имеющий ф-ции сортировки и поиска по заданному вектору

**ОПЕРАЦИИ:**

***Конструктор****:*

Вход: вектор типа Type.

Начальные значения: нет.

Процесс: инициализация данных.

Постусловие: создан вектор, заполненный случайными числами.

**Операция***: вывода на экран*

Вход: вектор

Предусловия:

вектор, заполненный случайными числами.

Процесс: цикл с выводом элементов на экран.

Выход: нет.

Постусловие: тот же вектор.

***Операция****: заполнения случайными числами*

Вход: вектор

Предусловия: вектор

Процесс: цикл с заполнением элементов вектора случайными числами.

Выход: вектор с новыми значениями.

Постусловие: новый вектор.

***Операция****: сортировка методом Шелла*

Вход: заполненный вектор

Предусловия: заполненный вектор

Процесс: цикл в цикле... меняющий местами элементы вектора Выход: отсортированный вектор.

Постусловие: отсортированный вектор.

***Операция****: сортировка методом слияния*

Вход: заполненный вектор

Предусловия: заполненный вектор

Процесс: цикл в цикле...и тд., меняющий местами элементы вектора

Выход: отсортированный вектор.

Постусловие: отсортированный вектор.

***Операция****: смены местами элементов*

Вход: 2 указателя на элементы

Предусловия: 2 указателя на элементы

Процесс: *смена местами элементов вектора*

Выход: вектор, поменял местами два своих элемента.

Постусловие: вектор

***Операция****: сортировка выбором*

Вход: заполненный вектор

Предусловия: заполненный вектор

Процесс: цикл в цикле...и тд., меняющий местами элементы вектора

Выход: отсортированный вектор.

Постусловие: отсортированный вектор.

***Операция****: нахождение минимального числа в векторе*

Вход: заполненный вектор

Предусловия: заполненный вектор

Процесс: нахождение минимального элемента

Выход: индекс минимального элемента.

Постусловие: вектор.

***Операция****: прямой поиск*

Вход: заполненный вектор

Предусловия: заполненный вектор

Процесс: цикл, сравнивающий ключ с элементов

Выход: индекс элемента.

Постусловие: вектор.

***Операция****: поиск числа фиббоначи*

Вход: заполненный вектор

Предусловия: заполненный вектор, порядок и ключ

Процесс: цикл в цикле...и тд., ищет число фиббоначи

Выход: индекс числа фиббоначи.

Постусловие: отсортированный вектор.

**КОНЕЦ АТД**

**Определение класса «MyVector»**

// this is task №1, my variant - the even

#include <iostream>

#include <vector>

#include <cstdlib>

#include <ctime>

#include <clocale>

using namespace std;

template <typename Type>

class MyVector : public std::vector<Type>

{

public:

MyVector(vector <Type> str) : std::vector<Type>(str){} //конструктор

~MyVector()// деструктор

{

cout << endl << " Object to destroyed!!!";

}

void print() // выводит вектор на экран

{

int i = 0;

auto it = begin();

while (it != end())

{

if (i > 0)

cout << " ";

cout << \*it;

++it;

++i;

}

cout << endl;

}

void rand\_fill() // заполняет вектор случайными числами

{

srand(time(0));

for (auto it = begin(); it != end(); it++)

{

\*it = rand() % 100000;

}

cout << endl;

}

void sort\_shell() // сортировка методом Шелла

{

int e = 0; // numbers obmenov

int comp = 0; // numbers sravneniy

int step = size() / 2;

while (step > 0)

{

++comp;

for (int i = 0; i < (size() - step); i++)

{

int j = i;

while (j >= 0 && at(j) > at(j + step))

{

++comp;

++comp;

int temp = at(j);

at(j) = at(j + step);

at(j + step) = temp;

j--;

++e;

++e;

}

}

step = step / 2;

}

cout << "Method by Shell e = " << e << " comp = " << comp;

}

void sort\_merge(int first, int last, static int &e, static int &comp) // сортировка методом слияния

{

e++;

if (first < last)

{

sort\_merge(first, (first + last) / 2, e, comp); сортир. левую сторону

sort\_merge((first + last) / 2 + 1, last, e, comp); //сорт. правую сторону

merge(first, last, e, comp); // меняет местами

}

}

void merge(int first, int last, static int &e, static int &comp)

{

int middle, start, final, j;

vector<Type> mas;

middle = (first + last) / 2;

start = first;

final = middle + 1;

for (j = first; j <= last; j++) {

if ((start <= middle) && ((final > last) || (at(start) < at(final))))

{

++comp;

++comp;

++comp;

mas.push\_back(at(start));

start++;

}

else

{

++comp;

++comp;

++comp;

mas.push\_back(at(final));

final++;

}

}

// return result

for (j = first; j <= last; j++)

{

at(j) = mas[j - first];

}

}

void swap\_my(iterator begin, iterator min, static int &e1, static int &comp1) // меняет местами элементы вектора

{

Type temp = \*begin;

\*begin =\*min;

\*min = temp;

++e1;

++e1;

++e1;

}

void sort\_case(static int &e1, static int &comp1) // сортировка выбором

{

for (auto it = begin(); it != end(); it++)

{

swap\_my(it, min(it, e1, comp1), e1, comp1);

}

}

iterator min(iterator it, static int &e1, static int &comp1) // ищет минимальное число в векторе

{

iterator min = it;

for (; it != end(); it++)

{

if (\*it < \*min) {

min = it;

++comp1;

}

}

return min;

}

int find\_direct(int number) //возвращает индекс числа, прямой поиск

{

int index = 0;

for (auto it = begin(); it != end(); it++, index++)

{

if (\*it == number)

return index;

}

return -1;

}

int number\_fib(const int n)// вычисляет число фиббоначи

{

unsigned long long f1, f2;

if ((n == 0) || (n == 1)) {

return n;

} else if (n >= 2) {

f1 = number\_fib(n - 1);

f2 = number\_fib(n - 2);

return f1 + f2;

}

}

int find\_fibonachi(const int n, const int key)// поиск числа фиббоначи

{

int t, search, mid, i, c;

int f1, f2;

int j = 1;

while (number\_fib(j) < (n + 1))

{

j = j + 1;

}

mid = n - number\_fib(j - 2) + 1;

f1 = number\_fib(j - 2);

f2 = number\_fib(j - 3);

c = 0;

for (;;)

{

if (c == 1)

break;

if (mid > 0)

if (key == at(mid))

break;

if ((mid <= 0) || (key > at(mid)))

{

if (f1 == 1)

{

c = 1;

}

else

{

mid = mid + f2;

f1 = f1 - f2;

f2 = f2 - f1;

}

}

else

{

if (f2 == 0)

{

c = 1;

}

else

{

mid = mid - f2;

t = f1 - f2;

f1 = f2;

f2 = t;

}

}

}

if (c)

search = 0;

else

search = mid;

if ((!search) || (search == n)) {

cout << ("number is not found ");

return -1;

}

else

return search;

}

};

**Реализация на matlab**

function MyVector()

mas = [12 8 9 1 4 5];

last = length(mas);

k = 0;

k = input('1) print \n2) sort\_shell \n3) sort\_merge \n4) cort\_case \n5) find\_fibonachi \n6) find\_direct \n7) rand\_fill \n Enter to 8 for exit\n');

while (k ~= 8)

Start = cputime();

if k == 1

myPrint(mas);

pause();

end

if k == 2

disp('Method Shell');

mas = sortShell(mas);

pause();

end

if k == 3

disp('Method merge');

mas = sort\_merge(mas);

pause();

end

if k == 4

disp('Method sort\_case');

mas = sortCase(mas);

pause();

end

if k == 5

n = input('enter to n: ');

key = input('nenter to key: ');

result = find\_fibonachi(n, key, mas) + 1

pause();

end

if k == 6

t = input('enter to key: ');

result = findDirection(mas, t)

pause();

end

if k == 7

mas = myRand(mas);

disp('random');

pause();

end

if k == 8

break;

end

Time = cputime() - Start

pause();

k = input('1) print \n2) sort\_shell \n3) sort\_merge \n4) cort\_case \n5) find\_fibonachi \n6) find\_direct \n7) rand\_fill \n Enter to 8 for exit\n');

end

end

function myPrint(mas) %вывод вектора на экран

mas

end

function mas = myRand(mas) %заполняет вектор случайными числами

mas = rand(1, length(mas));

end

function result = number\_fib(n) %вычисляет число фиббоначи

f1 = 0;

f2 = 0;

result = 0;

if (n == 0) || (n == 1)

result = n;

return;

else

if n >= 2

f1 = number\_fib(n - 1);

f2 = number\_fib(n - 2);

result = f1 + f2;

return

end

end

end

function mid = find\_fibonachi(n, key, mas)% ищет число фиббоначи в массиве

t = 0;

search = 0;

mid = 0;

i = 0;

c = 0;

f1 = 0;

f2 = 0;

j = 1;

s = 0;

while (number\_fib(j) < (n + 1))

j = j + 1;

end

mid = n - number\_fib(j - 2) + 1;

f1 = number\_fib(j - 2);

f2 = number\_fib(j - 3);

c = 0;

for s = 0:s>=0

if (c == 1)

break;

end

if (mid > 0)

if (key == mas(mid))

break;

end

end

if ((mid <= 0) || (key > mas(mid)))

if (f1 == 1)

c = 1;

else

mid = mid + f2;

f1 = f1 - f2;

f2 = f2 - f1;

end

else

if (f2 == 0)

c = 1;

else

mid = mid - f2;

t = f1 - f2;

f1 = f2;

f2 = t;

end

end

s = s + 1;

end

if (c == 1)

search = 0;

else

search = mid;

end

if ((search == 0) || (search == n))

disp('number is not found');

pause();

return;

else

return;

end

end

function mas = sortShell(mas)%сортировка Шелла

e = 0;

comp = 0;

size = numel(mas);

step = round(size/2);

while step > 0

comp = comp + 1;

for i = (step+1:size)

temp = mas(i);

j = i;

while (j >= step+1) && (mas(j-step) > temp)

mas(j) = mas(j-step);

j = j - step;

comp = comp + 1;

comp = comp + 1;

e = e + 1;

e = e + 1;

end

mas(j) = temp;

end

if step == 2

step = 1;

else

step = round(step/2.2);

end

end

e

comp

end

function mas = sort\_merge(mas) %сортировка слиянием

if numel(mas) <= 1

return

else

middle = ceil(numel(mas) / 2);

left = mas(1:middle);

right = mas(middle+1:end);

left = sort\_merge(left);

right = sort\_merge(right);

if left(end) <= right(1)

mas = [left right];

return

end

%merge(left,right)

c = 1;

while (numel(left) > 0) && (numel(right) > 0)

if(left(1) <= right(1))

mas(c) = left(1);

left(1) = [];

else

mas(c) = right(1);

right(1) = [];

end

c = c + 1;

end

if numel(left) > 0

mas(c:end) = left;

elseif numel(right) > 0

mas(c:end) = right;

end

end

end

function mas = sortCase(mas) %сортировка выбором

count = length(mas);

for i=1:(count - 1)

min = i;

for j =(i+1):count

if mas(min) > mas(j)

min = j;

end

end

if min~=i

t=mas(i);

mas(i)= mas(min);

mas(min)= t;

end

end

end

function index = findDirection(mas, number) %прямой поиск

index = 1;

it = 1;

for it = 1:length(mas)-1

if mas(it) == number

return;

else

index = index + 1;

end

end

end

**Тестирование**

**задание 1:**

#include <iostream>

#include "interface.h"

#include <vector>

using namespace std;

int

main()

{

vector <int> s;

srand(time(0));

for (long int i = 8; i > 0; i--) {

s.push\_back(rand() % 100);

}

s.push\_back(34);

// for merge

static int e = 0;

static int comp = 0;

static int \*ex = &e;

static int \*compan = &comp;

// for sort\_case

static int e1 = 0;

static int comp1 = 0;

static int \*ex1 = &e1;

static int \*compan1 = &comp1;

MyVector <int> object(s);

cout << "1) print \n2) sort\_shell \n3) sort\_merge \n4) cort\_case \n5) find\_fibonachi \n6) find\_direct \n7) rand\_fill \n Enter to q for exit\n" << endl;

char k;

cin >> k;

while (k != 'q')

{

cout << endl << "1) print \n2) sort\_shell \n3) sort\_merge \n4) cort\_case \n5) find\_fibonachi \n6) find\_direct \n7) rand\_fill \n Enter to q for exit\n" << endl;

long long start = clock();

switch (k)

{

case '1':

object.print();

break;

case '2':

object.sort\_shell();

break;

case '3':

object.sort\_merge(0, object.size() - 1, \*ex, \*compan);

cout << "Method merge \t e = " << e << " comp = " << comp << endl;

break;

case '4':

object.sort\_case(\*ex1, \*compan);

cout << "Method sort\_case \t e = " << e1 << " comp = " << comp1 << endl;

break;

case '5':

cout << "enter to n: ";

int n, key;

cin >> n;

cout << "\nenter to key: ";

cin >> key;

cout << object.find\_fibonachi(n, key) + 1 << endl;

break;

case '6':

int t;

cout << "enter to key: ";

cin >> t;

cout << object.find\_direct(t) + 1 << endl;

break;

case '7':

object.rand\_fill();

break;

}

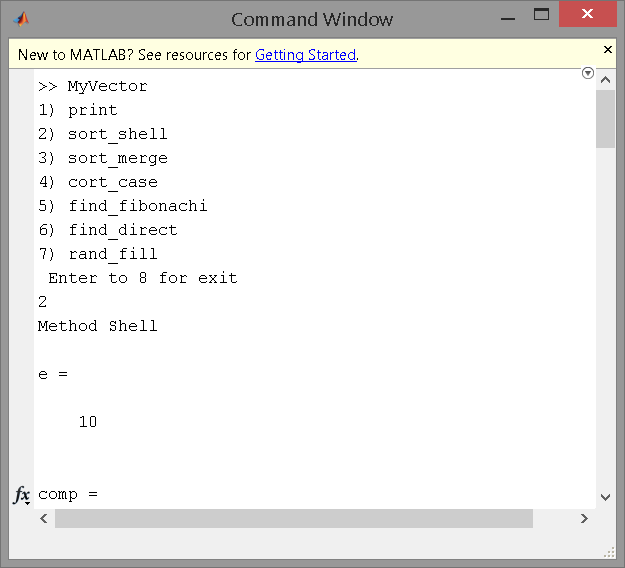
cout << " time = " << clock() - start << endl;

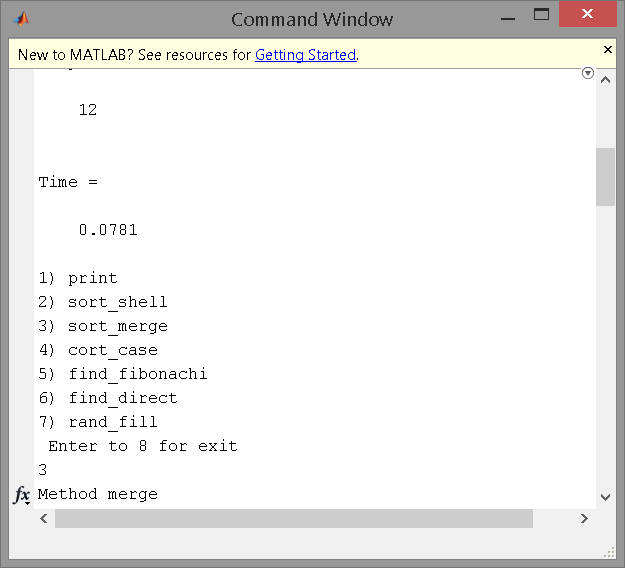
cin >> k;

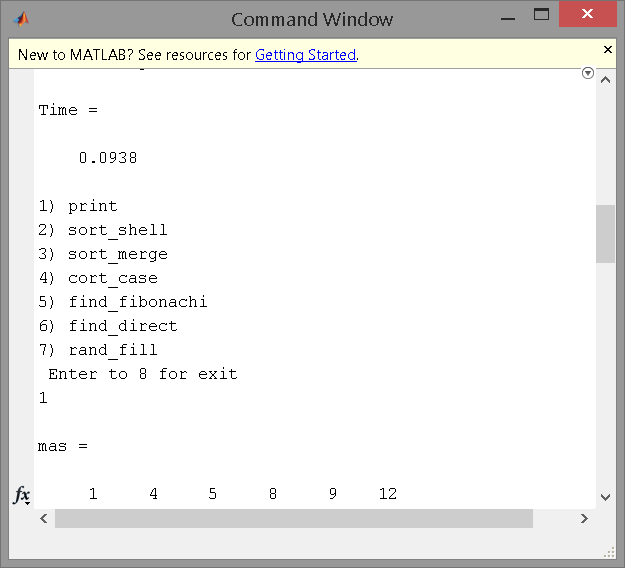
}

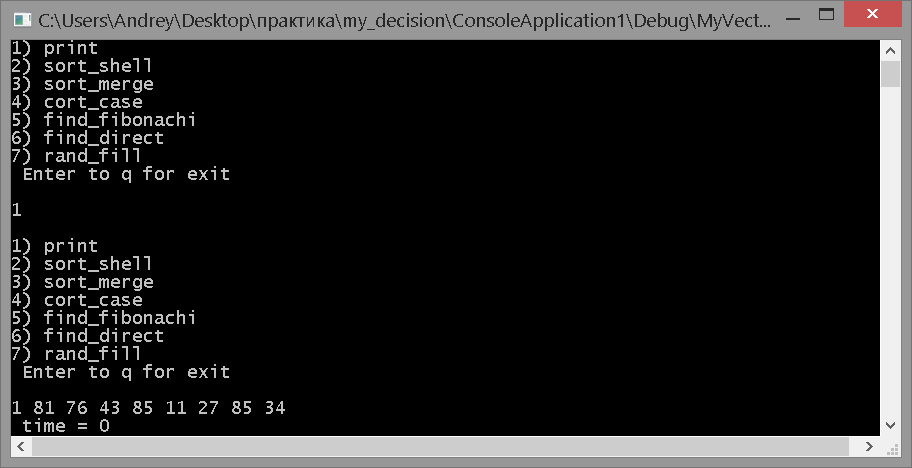
return 0;

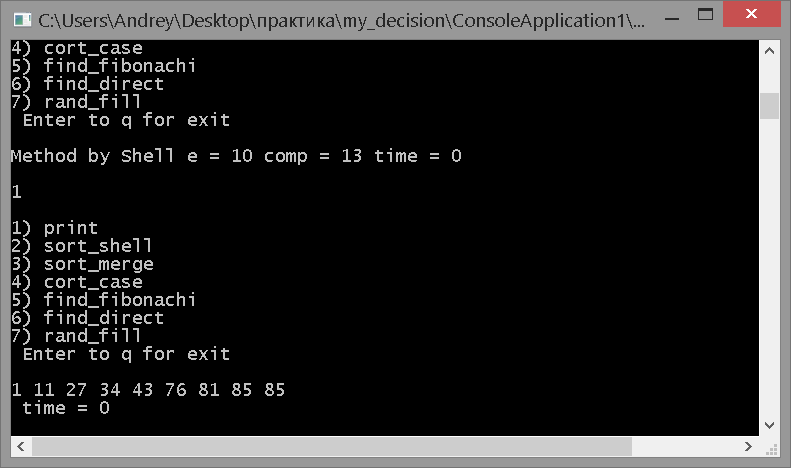
**}**

****

****

****

****

****

**Вывод:**

Вследствие проведенного тестирования были получены следующие данные о методах сортировки и поиска в массиве/векторе:

1. Самый эффективный метод сортировки из данных - метод сортировки выбором
2. Сортировка слиянием крайне неэффективна в небольших массивах
3. Сортировка Шелла эффективна в массивах, состоящих из больших чисел

## Задание 2. АТД Матрица

### Цель работы

1. Ознакомиться с операциями над матрицами.
2. Приобрести навыки разработки класса в с++.
3. Пробрести навыки обработки матриц в среде Matlab.

### Задание

1. Разработать АТД «Matrix», реализующего следующие операции:

* ввод матрицы из файла;
* вывод матриц в файл;
* ввод матрицы с клавиатуры;
* вывод матрицы на экран;
* операция сложения матриц;
* операция вычитания матриц;
* операция умножения матриц;
* транспонирование матрицы;
* операция, соответствующая варианту задания (Таблица 4.2.1).

2. Вариант индивидуального задания по обработке матриц:

Дана матрица размером n\*m. Найти первый столбец с минимальным элементом и

отсортировать строки матрицы по неубыванию значений элементов найденного столбца.

***Формат АТД***

**АТД Matrix**

Автоматический тип данных/шаблон (Type)

**ДАННЫЕ:**

матрицы типа Type.

Шаблон, имеющий ф-ции и операторы для работы с матрицами

**ОПЕРАЦИИ:**

*Конструктор:*

Вход: создает матрицу.

Начальные значения: количество строк, столбцов и значение.

Процесс: инициализация данных.

Постусловие: создана матрица, состоящая из чисел

**Операция***: загрузки матрицы из файла*

Вход: файл

Предусловия:

матрица

Процесс: создание и запись матрицы

Выход: матрица.

Постусловие: матрица

***Операция****: запись матрицы в файл*

Вход: матрица

Предусловия: матрица

Процесс: запись матрицы в файл.

Выход: файл.

Постусловие: файл.

***Операция****: ввод матрицы с клавиатуры*

Вход: матрица

Предусловия: пуста

Процесс: поэлементная инициализация матрицы

выход: матрица

Постусловие: матрица.

***Операция****: сортировка методом слияния*

Вход: заполненный вектор

Предусловия: заполненный вектор

Процесс: цикл в цикле...и тд., меняющий местами элементы вектора

Выход: отсортированный вектор.

Постусловие: отсортированный вектор.

***Операция****: очистка памяти матрицы*

Вход: матрица

Предусловия: матрица

Процесс: *очитка памяти*

Выход: пустая матрица

Постусловие: матрица

***Операция****: транспонирования матрицы*

Вход: матрица

Предусловия: полная

Процесс: транспонирование

Выход: транспонированная матрица.

Постусловие: матрица.

***Операция****: индивидуальное задание вариант 14*

Вход: матрица

Предусловия: матрица

Процесс: ищет первый столбец с минимальным элементом и сортирует строки матрицы по неубыванию значений элементов найденного столбца.

Выход: матрица

Постусловие: матрица.

***Операция****: получение значения элемента матрицы*

Вход: матрица

Предусловия: матрица

Процесс: *получение значения элемента матрицы*

Выход: значение элемента.

Постусловие: значение элемента.

***Операция****: присвоить значение элементу матрицы*

Вход: матрица

Предусловия: матрица

Процесс: присвоить значение элементу матрицы

Выход: нет.

Постусловие: нет.

**КОНЕЦ АТД**

**определение класса «Matrix»**

#include <ostream>

#include <iostream>

#include <string>

#include <ctime>

#include <cstring>

#include <fstream>

using namespace std;

template<typename \_Type>

//#define \_Type int

class Matrix {

typedef std::vector<\_Type> MatrixLine;

typedef std::vector<MatrixLine> MatrixValue;

typedef Matrix MatrixT;

MatrixValue \_value;

public:

Matrix(int rows, int cols, \_Type def) // конструктор

{

clear(rows, cols, def);

}

void LoadByFile(string file) //загрузка матрицы из файла

{

\_value.clear();

ifstream s = ifstream(file);

string in;

int len = 1;

getline(s, in);

for each (char c in in)

if (c == ' ') len++;

s.seekg(0);

while (!s.eof())

{

MatrixLine line;

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

\_Type h;

s >> h;

line.push\_back(h);

}

\_value.push\_back(line);

}

s.close();

}

void SaveFile(string file) //запись матрицы в файл

{

ofstream s(file);

for each (auto line in \_value)

{

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

s << line[i];

if (i != getColCount() - 1)

s << ' ';

}

s << endl;

}

s.close();

}

static MatrixT ReadKeyboard() //чтение матрицы с клавиатуры

{

int x, y;

cout << "Введите размер м-ци: (2 числа через пробел)\n";

cin >> x >> y;

MatrixT m(x, y, 0);

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

cout << "Введите " << i + 1 << "строку (" << y << " чисел через пробел):\n";

for (int j = 0; j < y; j++)

cin >> m.\_value[i][j];

}

return m;

}

void clear(int rows, int cols, \_Type def) //очистка памяти матрицы

{

\_value.clear();

for (auto i = 0; i < rows; i++) {

auto line = MatrixLine(cols);

std::fill(line.begin(), line.end(), def);

\_value.push\_back(line);

}

}

bool operator==(const MatrixT &other) //оператор сравнения ==

{

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

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

if (get(i, j) != other.get(i, j))

return false;

}

}

return true;

}

bool operator>=(const MatrixT &other) // оператор сравнения >=

{

return determinant() >= other.determinant();

}

bool operator<=(const MatrixT &other) // оператор сравнения <=

{

return determinant() <= other.determinant();

}

bool operator!=(const MatrixT &other) // оператор неравенства !=

{

return !operator==(other);

}

Matrix operator+(const MatrixT &that) // оператор сложения матриц

{

if (getRowCount() != that.getRowCount() ||

getColCount() != that.getColCount()) {

return \*this;

}

Matrix result(getRowCount(), getColCount(), 0);

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

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

result.put(i, j, get(i, j) + that.get(i, j));

}

}

return result;

}

Matrix operator-(const MatrixT &that) //оператор вычитания матриц

{

if (getRowCount() != that.getRowCount() ||

getColCount() != that.getColCount()) {

return \*this;

}

Matrix result(getRowCount(), getColCount(), 0);

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

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

result.put(i, j, get(i, j) - that.get(i, j));

}

}

return result;

}

Matrix operator\*(\_Type number) // оператор умножения матрицы на число

{

Matrix result(getRowCount(), getColCount(), 0);

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

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

result.put(i, j, get(i, j) \* number);

}

}

return result;

}

Matrix operator\*(const MatrixT &that) //оператор переумножения матриц

{

Matrix result(getRowCount(), that.getColCount(), 0);

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

int value = 0;

for (int j = 0; j < that.getColCount(); j++) {

for (int k = 0; k < getColCount(); k++) {

value += get(i, k) \* that.get(k, j);

result.put(i, j, value);

}

value = 0;

}

}

return result;

}

Matrix invert() //обратная матрица

{

int size = getRowCount();

if (size != getColCount()){

return \*this;

}

Matrix result(size, size, 0);

\_Type det = determinant();

if (det == 0)

{

return \*this;

}

\_Type temp;

bool \_invert = false;

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

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

temp = \_invert ? 0 : 1;

temp = minor(i, j);

temp /= det;

result.put(i, j, temp);

\_invert = !\_invert;

}

}

result = result.transposition();

return result;

}

MatrixT speed()

{

return \*this;

}

\_Type determinant() const //определитель матрицы

{

\_Type result = 0;

int len = getRowCount();

if (len == getColCount()) {

int x;

\_Type line;

for (int invert = 0; invert < 2; invert++){

for (int offest = 0; offest < len; offest++) {

line = 1;

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

x = offest + i;

if (invert){

x = -x;

while (x < 0)

x += len;

}

else {

while (x >= len)

x -= len;

}

line \*= get(x, i);

}

result += invert ? -line : line;

}

}

}

return result;

}

/\*///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////\*/

bool isDegenerate() const

{

return false;

}

bool isSpeed() const

{

return false;

}

bool isIdentity() const

{

return false;

}

bool isTriangular() const

{

return false;

}

/\*///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////\*/

bool isSymmetric() const //проверка на симметричность матриц

{

int len = getRowCount();

if (!len || len != getColCount())

return false;

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

for (int k = i + 1; k < len; k++) {

if (get(i, k) != get(k, i))

return false;

}

}

return true;

}

int getRowCount() const // получить количество строк

{

return \_value.size();

}

int getColCount() const //получить количество столбцов

{

if (\_value.empty())

return 0;

return \_value[0].size();

}

MatrixT transposition() //транспонирование матрицы

{

MatrixT mtx = MatrixT(getColCount(), getRowCount(), 0);

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

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

mtx.put(j, i, get(i, j));

}

}

return mtx;

}

Matrix minCol() // Дана матрица размером n\*m. Найти первый столбец с минимальным элементом и отсортировать строки матрицы по неубыванию значений элементов найденного столбца.

{

MatrixT mtx = MatrixT(getRowCount(), getColCount(), 0);

\_Type min = get(0, 0);

int jmin;

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

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

mtx.put(i, j, get(i, j));

if (\_value[i][j] < min) {

min = \_value[i][j];

jmin = j;

}

}

}

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

for (int m = 0; m < getRowCount() - i - 1; m++) {

if (mtx.get(m, jmin) > mtx.get(m + 1, jmin))

{

MatrixLine t = mtx.\_value[m];

mtx.\_value[m] = mtx.\_value[m + 1];

mtx.\_value[m + 1] = t;

}

}

}

return mtx;

}

\_Type minor(int row, int col) const // нахождение минора матрицы

{

Matrix my(\_value.size() - 1, \_value.size() - 1, 0);

for (int i = 0, i2 = 0; i < my.\_value.size(); i++, i2++) {

for (int j = 0, j2 = 0; j < my.\_value.size(); j++, j2++) {

if (i == row) i++;

if (j == col) j++;

my.put(i2, j2, \_value[i][j]);

}

}

return my.determinant();

}

\_Type get(int row, int col) const // получить значение элемента матрицы

{

if ((unsigned int)row >= \_value.size())

return 0;

auto line = \_value[row];

if ((unsigned int)col >= line.size())

return 0;

return line[col];

}

void put(int row, int col, \_Type value) // присвоить значение элементу матрицы

{

if (row < \_value.size()){

auto line = \_value[row];

if (col < line.size())

\_value[row][col] = value;

}

}

private:

template<typename U> friend std::ostream& operator<<(std::ostream& os, const Matrix<U>& m);

};

template<typename \_Type>

std::ostream &operator<<(std::ostream &output, const Matrix<\_Type> &m) // оператор вывода матрицы

{

if (m.getRowCount()) {

for each (auto line in m.\_value) {

for each (auto item in line)

std::cout << item << ' ';

std::cout << std::endl;

}

}

return output;

}

#endif /\*\* \_MATRIX\_H\_ \*/

**Реализация на матлаб**

function matrix()

a = [4 5 6; 1 2 3; 7 8 9]

[b, i, j] = inputFile('matrix1.txt')

outputFile(a, 3, 3, 'matrix2.txt')

key = input('Меню:\n1)ввод матрицы с клавиатуры\n2)сложение матриц\n3)вычитание матриц\n4)умножение матриц\n5)транспонирование матрицы\n6)Индивидуальное задание(вариант №14):\n7)для выхода\n')

while (key > 0)

if key == 1

a = ReadKeyboard();

outputFile(a, length(a), length(a), 'matrix2.txt')

end

if key == 2

fprintf('матрица');

a

fprintf('матрица');

b

if length(a) == length(b)

disp('сумма матриц c равна:');

c = a + b

else

disp('ОШИБКА! матрицы должны быть одинакового размера')

key = input('Меню:\n1)ввод матрицы с клавиатуры\n2)сложение матриц\n3)вычитание матриц\n4)умножение матриц\n5)транспонирование матрицы\n6)Индивидуальное задание(вариант №14):\n7)для выхода\n')

pause();

continue;

end

end

if key == 3

fprintf('матрица');

a

fprintf('матрица');

b

if length(a) == length(b)

disp('сумма матриц c равна:');

c = a + b

else

disp('ОШИБКА! матрицы должны быть одинакового размера')

key = input('Меню:\n1)ввод матрицы с клавиатуры\n2)сложение матриц\n3)вычитание матриц\n4)умножение матриц\n5)транспонирование матрицы\n6)Индивидуальное задание(вариант №14):\n7)для выхода\n')

pause()

continue;

end

end

if key == 4

fprintf('матрица');

a

fprintf('матрица');

b

disp('произведение матриц c равна:');

c = a \* b

end

if key == 5

fprintf('матрица');

a

disp('транспонированная матрица равна:');

a'

end

if key == 6

fprintf('матрица');

a

result = minCol(a, 3, 3)

end

if key == 7

break;

end

pause()

key = input('Меню:\n1)ввод матрицы с клавиатуры\n2)сложение матриц\n3)вычитание матриц\n4)умножение матриц\n5)транспонирование матрицы\n6)Индивидуальное задание(вариант №14):\n7)для выхода\n')

end

end

function print(matrix)

matrix

end

function c = ReadKeyboard()

c = input('Введите матрицу:');

disp('Ваша матрица: ');

print(c);

end

function c = read(file)

[f,c] = fopen(file, 'rt');

if f ~=-1

c = ' ';

while feof(f) == 0

line = fgetl(f);

c = char(c, line);

end

fclose(f);

end

end

function [ c, n, m ] = inputFile(file)

f = fopen(file, 'rt');

n = fscanf(f,'%d',1);

m = fscanf(f,'%d',1);

for i=1:n

for j=1:m

c(i,j) = fscanf(f,'%f',1);

end

end

fclose(f);

end

function outputFile( c, n, m, file)

f = fopen(file, 'wt');

for i=1:n

for j=1:m

fprintf(f,'%f\t',c(i,j));

end

fprintf(f,'\n');

end

fclose(f);

end

function c = minCol(c, n, m)

jmin = min(min(c))

i = 1;

k = 1;

t = 0;

for k=1:k<m-1

for i=1:i<=n-i-1

if c(i,jmin) > c(i+1,jmin)

t=c(i,:);

c(i,:)=c(i+1,:);

c(i+1,:)=t;

end

end

end

end

**Тестирование**

**Задание 2:**

#include <iostream>

#include <string>

#include <ctime>

#include <cstring>

#include "matrix.h"

using namespace std;

void // проверка на симметричность

\_\_test\_isSymmetric()

{

Matrix<int> m(4, 4, 0);

std::cout << "Матрица: " << std::endl << m;

std::cout << "Симметрична: " << (m.isSymmetric() ? "Нет" : "Да") << std::endl << std::endl;

}

void //сложение матриц

\_\_test\_operatorPlus()

{

Matrix<int> a(3, 3, 2);

Matrix<int> b(3, 3, 3);

std::cout << "Матрица A:" << std::endl << a;

std::cout << "Матрица B:" << std::endl << b;

std::cout << "Матрица A+B:" << std::endl << a + b << std::endl;

}

void // вычитание матриц

\_\_test\_operatorMinus()

{

Matrix<int> a(3, 3, 2);

Matrix<int> b(3, 3, 3);

std::cout << "Матрица A:" << std::endl << a;

std::cout << "Матрица B:" << std::endl << b;

std::cout << "Матрица A-B:" << std::endl << a - b << std::endl;

}

void //обратная матрица

\_\_test\_invert()

{

Matrix <int> m(3, 3, 0);

m.put(0, 0, 5);

m.put(1, 1, 3);

m.put(2, 2, 2);

std::cout << m.invert() << std::endl;

}

void //транспонирование матрицы

\_\_test\_transposition()

{

Matrix <int> a(8, 4, 1);

a.put(5, 1, 2);

a.put(2, 3, 1);

a.put(2, 2, 2);

std::cout << "Матрица: " << std::endl << a << std::endl;

std::cout << "Transp.: " << std::endl << a.transposition() << std::endl;

}

void //определитель матрицы

\_\_test\_determinant()

{

Matrix <int> a(3, 3, 0);

a.put(0, 0, 5);

a.put(1, 1, 3);

a.put(2, 2, 2);

std::cout << "Матрица A:" << std::endl << a;

std::cout << "det(A)=" << a.determinant() << std::endl;

}

void //минор матрицы

\_\_test\_isMinor()

{

Matrix <int> m(4, 4, 0);

for (int i = 0, i2 = 0; i < m.getRowCount(); i++, i2++) {

for (int j = 0, j2 = 0; j < m.getColCount(); j++, j2++) {

m.put(i2, j2, m.get(i, j));

}

}

int row = 3, col = 3;

std::cout << "Матрица: " << std::endl << m;

std::cout << "Минор (" << row << "," << col << ") = " << m.minor(row, col) << std::endl << std::endl;

}

void //переумножение матриц

\_\_test\_operatorUMN()

{

Matrix<int> a(3, 3, 2);

Matrix<int> b(2, 2, 3);

std::cout << "Матрица A:" << std::endl << a;

std::cout << "Матрица B:" << std::endl << b;

std::cout << "Матрица A\*B:" << std::endl << a\*b << std::endl;

}

void //сравнение матриц

\_\_test\_equality()

{

Matrix<int> a(3, 3, 3);

Matrix<int> b(3, 3, 2);

std::cout << "Матрица A:" << std::endl << a;

std::cout << "Матрица B:" << std::endl << b << std::endl;

std::cout << "A=B ? " << (a == b ? "Да" : "Нет") << std::endl;

std::cout << "A!=B ? " << (a != b ? "Да" : "Нет") << std::endl;

std::cout << "A>=B ? " << (a >= b ? "Да" : "Нет") << std::endl;

std::cout << "A<=B ? " << (a <= b ? "Да" : "Нет") << std::endl << std::endl;

}

void //загрузка матрицы из файкла

\_\_test\_LoadByFile()

{

Matrix <int> m(3, 3, 0);

m.LoadByFile("matrix2.txt");

}

void //запись матрицы в файл

\_\_test\_SaveFile()

{

Matrix <int> m(3, 3, 0);

m.SaveFile("matrix2.txt");

}

void // Дана матрица размером n\*m. Найти первый столбец с минимальным элементом и отсортировать строки матрицы по неубыванию значений элементов найденного столбца.

\_\_test\_minCol()

{

Matrix<int> m(5, 4, 0);

cout << "Индивидуальное задание (вариант №14):" << endl;

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

for (int j = 0; j < m.getColCount(); j++) {

m.put(i, j, rand() % 100 - 50);

}

}

cout << m << endl << "---" << endl << m.minCol() << endl;

}

int //главная ф-ция (меню)

main(int argc, char \*\*argv)

{

setlocale(0, "Russian");

char key;

cout << "Меню:\n1)ввод матрицы с клавиатуры\n2)сложение матриц\n3)вычитание матриц\n4)умножение матриц\n5)транспонирование матрицы\n6)Индивидуальное задание(вариант №14):\n'q' для выхода\n";

cin >> key;

while (key != 'q')

{

switch (key)

{

case '1':

cout << "\nВаша матрица: \n" << Matrix<int>::ReadKeyboard() << endl;

break;

case '2':

cout << "\nВвод матрицы из файла..." << endl;

cout << "Вывод матриц в файл..." << endl;

\_\_test\_operatorPlus();

break;

case '3':

cout << "\nВвод матрицы из файла..." << endl;

cout << "Вывод матриц в файл..." << endl;

\_\_test\_operatorMinus();

break;

case '4':

cout << "\nВвод матрицы из файла..." << endl;

cout << "Вывод матриц в файл..." << endl;

\_\_test\_operatorUMN();

break;

case '5':

cout << "\nВвод матрицы из файла..." << endl;

cout << "Вывод матриц в файл..." << endl;

\_\_test\_transposition();

break;

case '6':

cout << "\nВвод матрицы из файла..." << endl;

cout << "Вывод матриц в файл..." << endl;

\_\_test\_minCol();

break;

}

system("pause");

system("cls");

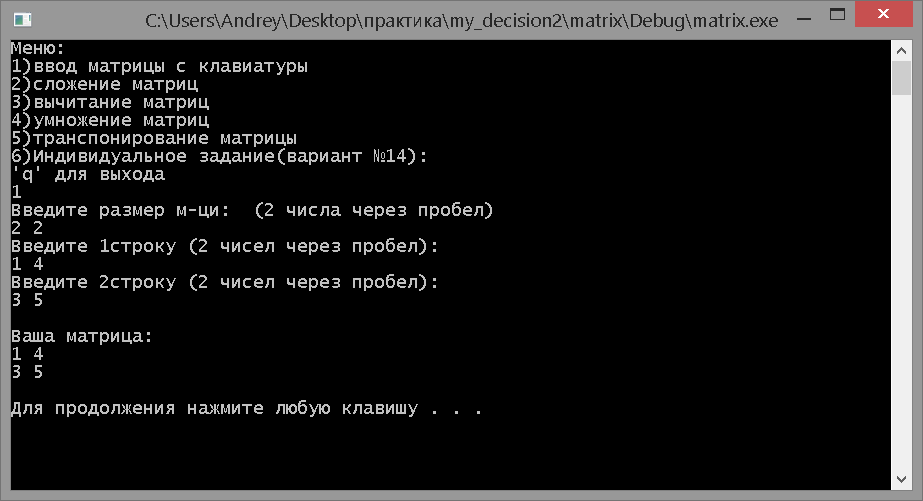
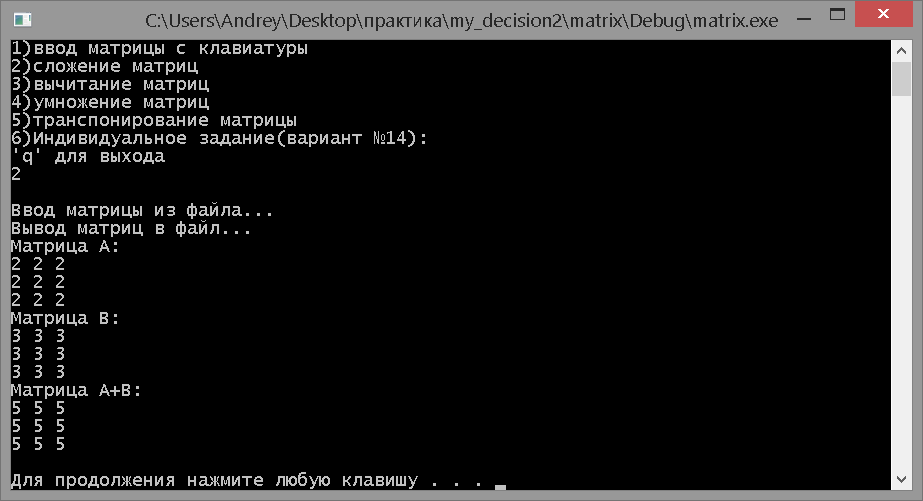
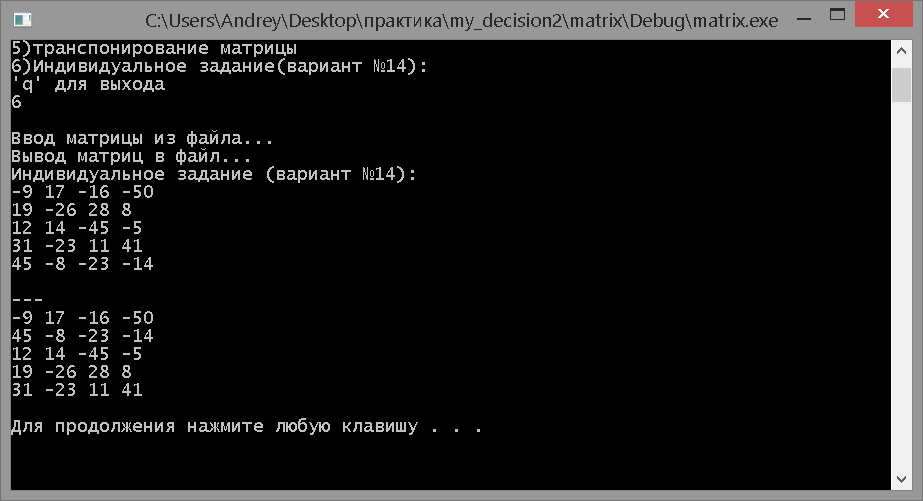
cout << "Меню:\n1)ввод матрицы с клавиатуры\n2)сложение матриц\n3)вычитание матриц\n4)умножение матриц\n5)транспонирование матрицы\n6)Индивидуальное задание(вариант №14):\n'q' для выхода\n";

cin >> key;

}

return EXIT\_SUCCESS;

**}**

**  **

**ВЫВОД:**

Вследствие проведенных тестов, я пришел к выводу, что реализация АТД "MATRIX" на якзыке MATLAB очень эффективна и проста, так как матлаб имеет большое количество встроенных ф-ций и операторов по работе с матрицами, в отличие от Си/Си++, не имеющего никаких либо встроенных средств по работе с матрицами :(

## Задание 3. Графика в Matlab

### Цель работы

1.Изучить основные графические возможности системы Matlab.

2.Создать программы, реализующие графический вывод данных.

### Задание

1.Составить и отладить программу для вывода графиков функций *f1*, *f2*, *f3*на основании задания из таблицы 2. Вывод графиков должен быть осуществлен в одном окне, графики должны быть подписаны, отмасштабированы.

Таблица 2 – Вариант задания по построению графиков функций

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Номер варианта** |  |  |  |  |
| 14 |  |  |  |  |

2.Составить и отладить программу для вывода графика трехмерной поверхности для функции *f4*() задания из таблицы 2.

3. Подготовить обзорный реферат по теме «Графические средства Matlab» и разработать программу, демонстрирующую эти возможности в соответствии с вариантом

Таблица 3 – Тема реферата

|  |  |
| --- | --- |
| **Номер варианта** | **Тип графика** |
| 14 | Графики векторов |

**Реализация на матлаб**

**f1,f2,f3:**

clear

x=0:0.1\*pi:2\*pi;

y1=sin(x) + cos(x);

y2=cos(x).\*x.^2;

y3=log(x).\*x.^2;

xmin=0;

xmax=2\*pi;

ymin=min([min(y1),min(y2), min(y3)])

ymax=max([max(y1),max(y2), max(y3)])

plot(x,y1,'b\*-', x, y2,'rd-.', x, y3,'g+--');

h=legend('sin(x) + cos(x)','cos(x).\*x.^2','log(x).\*x.^2',0);

grid on;

axis ( [xmin xmax ymin ymax])

set(gca,'FontName','Arial Cyr');

xlabel('X', 'Fontsize', 16); ylabel('Y', 'Fontsize', 16);

title('Функции', 'Fontsize', 16);

****

**График трехмерной поверхности для функции**

**f4:**

x=-20:0.2:20; y=-20:0.2:20;

[X,Y]=meshgrid(x,y);

r = (X.^2 + Y.^2).^0.5

Z= r + (cos(r)./r).^2

mesh(X,Y,Z)

title('r + (cos(r)./r).^2', 'Fontsize', 16);

****

**Вывод:**

Вследствие тестирования графической системы матлаб был изучен механизм работы его графического движка и сделан вывод, что матлаб является очень удобным средством работы с графиками, так как имеет огромное количество встроенных ф-ций для работы с 2D и 3D графикой.

**Заключение**

Вследствие проведенной практики были изучены методы работы с рабочей средой матлаб и проведены исследования возможностей матлаб, при работе с математическими выражениями и графической составляющей, встроенной в среду.

В частности к плюсам матлаба можно отнести, что он предоставляет пользователю большое количество (несколько сотен) функций для анализа данных, покрывающие практически все области математики, в частности:

1)Матрицы и линейная алгебра — алгебра матриц, линейные уравнения, собственные значения и вектора, сингулярности, факторизация матриц и другие.

2)Многочлены и интерполяция — корни многочленов, операции над многочленами и их дифференцирование, интерполяция и экстраполяция кривых и другие.

3)Математическая статистика и анализ данных — статистические функции, статистическая регрессия, цифровая фильтрация, быстрое преобразование Фурье и другие.

4) Обработка данных — набор специальных функций, включая построение графиков, оптимизацию, поиск нулей, численное интегрирование (в квадратурах) и другие.

5)Дифференциальные уравнения — решение дифференциальных и дифференциально-алгебраических уравнений, дифференциальных уравнений с запаздыванием, уравнений с ограничениями, уравнений в частных производных и другие.

6)Разреженные матрицы — специальный класс данных пакета MATLAB, использующийся в специализированных приложениях.

7)Целочисленная арифметика — выполнение операций целочисленной арифметики в среде MATLAB.

При тестировании выяснилось, что АТД позволяет создавать большие системы на основе разных уровней абстракции, от существующих в компьютере машинных инструкций, до разнообразных возможностей языка программирования, вплоть до сортировки, поиска и другой функциональности высокого уровня

Абстрактные типы данных — это лишь этап в истинной бесконечности создания все более и более мощных абстрактных механизмов, в чем и заключается суть эффективного использования компьютеров для решения современных задач.

Также были получены следующие данные о методах сортировки и поиска в массиве/векторе:

1. Самый эффективный метод сортировки из данных - метод сортировки выбором
2. Сортировка слиянием крайне неэффективна в небольших массивах
3. Сортировка Шелла эффективна в массивах, состоящих из больших чисел

Это все делает матлаб востребованным у программистов, работающих со сложными математическими структурами данных

**Список используемой литературы:**

1. Ашарина И.В. Объектно-ориентированное программирование вС++. Учебное пособие – М.: Горячая линия – Телеком, 2012 – 320с. (Эл.библиотека IPRbooks).
2. Вирт Н. Алгоритмы и структуры данных: Новая версия для Оберона + CD: Пер. с англ. — М.: ДМК Пресс, 2010. — 272 с. (НТБ – 2 экз.) - ISBN 978-5-94074-584-6
3. Гагарина Л. Г., Колдаев В. Д. Алгоритмы и структуры данных: Учеб.пособие для вузов. — М.: Финансы и статистика, 2009. — 303 с. (13 назв.) ISBN 978-5-279-03351-5
4. Кнут Д. Э. Искусство программирования – М.: Вильямс, 2010. – 832 с. (НТБ – 2 экз.), ISBN 978-5-8459-0080-7
5. Лафоре Р.  Основные алгоритмы. Объектно–ориентированное программирование вС++ – СПб.: Питер, 2006. – 924 с. (НТБ – 6 экз.), ISBN 5-94723-302-9