

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА - Российский технологический университет»**

**РТУ МИРЭА**

Институт искусственного интеллекта

Кафедра математического обеспечения и стандартизации

информационных технологий

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 4**

**Структуризация многоэлементных структур средствами struct** «СТРУКТУРЫ И АЛГОРИТМЫ ОБРАБОТКИ ДАННЫХ»

|  |  |
| --- | --- |
| Выполнил студень группы ИНБО-02-21      Принял старший преподаватель | Юдов С.А. |
| кафедры МОСИТ | Скворцова Л.А. |
| Практическая работа выполнена «\_\_»\_\_\_\_\_\_\_2022г. | \_\_\_\_\_\_\_\_\_ |
|  |  |
| «Зачтено» «\_\_»\_\_\_\_\_\_\_2022г. | \_\_\_\_\_\_\_\_\_ |
|  |  |

Москва 2022

**Содержание**

[Задание 1 3](#_Toc99285529)

[1. Условие задачи и варианта 3](#_Toc99285530)

[2. Разработка задачи 3](#_Toc99285531)

[3. Определение функций 4](#_Toc99285532)

[4. Реализация функций 4](#_Toc99285533)

[5. Кодирование алгоритма программы 5](#_Toc99285534)

[6. Таблица тестов программы 6](#_Toc99285535)

[Задание 2 7](#_Toc99285536)

[1. Определение функций 7](#_Toc99285537)

[2. Декомпозиция 7](#_Toc99285538)

[3. Реализация функций 7](#_Toc99285539)

[4. Кодирование алгоритма программы 8](#_Toc99285540)

[5. Таблица тестов программы 9](#_Toc99285541)

[Вывод 10](#_Toc99285542)

[Список информационных источников 11](#_Toc99285543)

# Задание 1

## 1. Условие задачи и варианта

* 1. Заказ и доставка товаров из супермаркета. Структура записи о товаре: название, артикул, количество, цена за единицу. Структура записи о заказе: номер заказа, дата заказа, фамилия заказчика, номер телефона, список товаров)
     1. Определить стоимость покупки по заказу.
     2. Добавить в список заказа новый товар.
     3. Добавить новый заказ в список заказов супермаркета.
     4. Удалить заказ с заданным номером из списка заказов супермаркета.

## 2. Разработка задачи

2.1. Постановка задачи.

2.1.1. Дано. Пустая страница

2.1.2. Результат. Таблица, заполненная записям, соответствующие структуре описанной в задании

2.2. Описание модели решения

Для описания модели решения рассмотрим структуры, которые будем использовать. Для структуры записи в задании четко прописаны необходимые поля и с этим все понятно. Подумаем насчёт наполнения структуры таблицы. Структура таблицы будет обладать полем для хранения количества имеющихся записей, “контейнером” (статическим массивом для задания 1) для хранения самих записей. Также данная структура будет обладать функционалом, описанным в задании под пунктом операции, то есть будут методы определения стоимости покупки по заказу, добавления в список заказа новый товар, добавление нового заказа в список заказов супермаркета, удаление заказа с заданным номером из списка заказов супермаркета. Для ясности структуры записей товара и заказов продемонстрированы на рисунках 1 и 2.

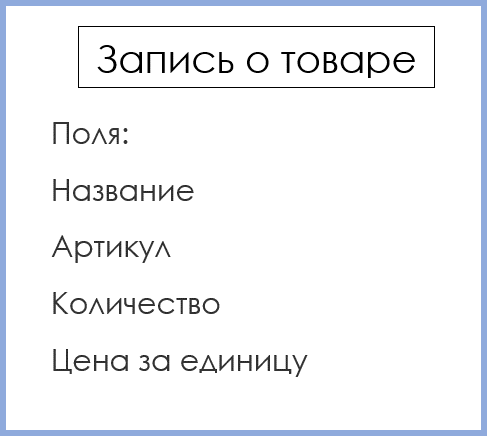


Рисунок 1 – Структура записи о товаре

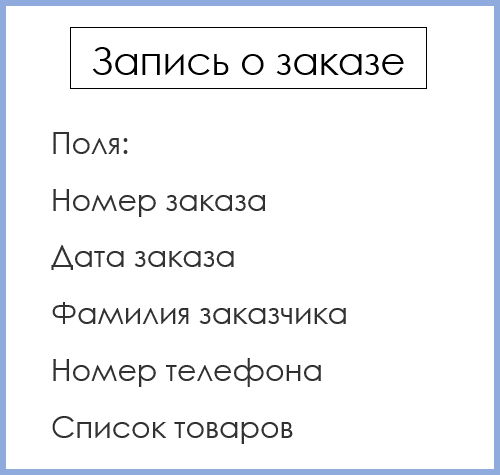


Рисунок 2 – Структура записи о заказе

А для ясности структуры таблиц записей товара и заказов продемонстрированы на рисунках 3 и 4.

После представления данных опишем сам ход работы. Изначально создается таблица товаров, после чего в неё добавляются записи с помощью имеющегося метода. Затем создаётся таблица заказов в которую добавляются записи о заказах супермаркета из имеющихся товаров.

2.3 Определение структуры на языке программирования

Структура записи.

struct RezEx

{

unsigned int CodStudent; // функция заполнения кода студента;

unsigned int CodDiscpline; // индекс из справочника

unsigned int scEx; // не больее 50

unsigned int scAc; // не больее 25

unsigned int mark; // 2 - 5

void convMark(); // функция по переводу баллов в оценку

};

Структура таблицы.

struct Table

{

unsigned int n = 0;

RezEx table[N];

void PrintTable(); // вывод таблицы

void Input(); // ввод записи

void Add\_Exsc(); //добавление экзам. балла

void ChetMark(); //подсчёт итоговых оценок

RezEx\* MakeVedomost(int& k); //сформировать ведомость по

дисциплине

};

2.3. Декомпозиция – список алгоритмов, которые требуются разработать в соответствии исследованной моделью

2.3.2. Определение прототипов функций:

1. Создание массива слов из строки

Предусловие. char \*\*arr – указатель на статичный массив, char \*str – указатель на строку, char \*s – указатель на пробельный символ

Постусловие. Заполненный массив из k элементов

int arrсreate(char \*\*arr, char \*str, char \*s)

1. Поиск и вывод слов-анаграмм

Предусловие char \*\*arr – указатель на статичный массив, int k > 0 – количество

Постусловие. Вывод слов-анаграмм

void checkingallarray(char \*\*arr, int k)

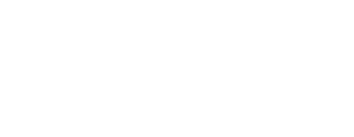
## 3. Определение функций

3.1. Создание массива слов из строки:

int



A



char \*\*arr

char \*str

char \*s



Входные данные



Результат

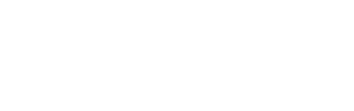


int arrсreate

3.2. Поиск и вывод слов-анаграмм



A



char \*\*arr

int k



Входные данные



Результат



void checkingallarray

## 4. Реализация функций

int arrсreate(char \*\*arr, char \*str, char \*s) {

int i = 0;

while (arr[i] != NULL) // проход по строке

{

// puts(arr[i]); // вывод слова

arr[++i] = strtok(NULL, s); // запись слова в двумерный массив

}

return i; // возвращение количества переменных в массиве

}

void checkingallarray(char \*\*arr, int k) {

for (int i = 0; i < k; i++) { // проход по элементам массива

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

char temp1[strlen(arr[i])];

strcpy(temp1, arr[i]);

char temp2[strlen(arr[j])];

strcpy(temp2, arr[j]);

sort(temp1, temp1 + strlen(temp1) \* sizeof(char));

sort(temp2, temp2 + strlen(temp2) \* sizeof(char));

if (strcmp(temp1, temp2) == 0 && i != j) {

printf("% s ", arr[i]);

}

}

}

}

## 5. Кодирование алгоритма программы

#include <iostream>

#include <cstdio>

#include <string>

#include <cstring>

#include <algorithm>

using namespace std;

int main() {

setlocale(LC\_ALL, "rus");;

char str[100] = "";

char \*arr[100];

char s[] = " ";

int k = 0;

cin.clear();

gets(str);

arr[0] = strtok(str, s); // инициализация массива слов с разделителем - пробелом

k = arrсreate(arr, str, s);

checkingallarray(arr, k);

return 0;

}

## 6. Таблица тестов программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер задачи | Исходные данные | Ожидаемый результат | Результат программы | Тест пройден/не пройден |
| 1 | cat tac ssa ddd | cat tac | cat tac | Тест пройден |
| 2 | man www nam | man nam | man nam | Тест пройден |

# Задание 2

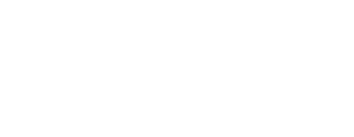
## 1. Определение функций

1.1. Создание массива слов из строки:

vector<string>



A



string str



Входные данные



Результат



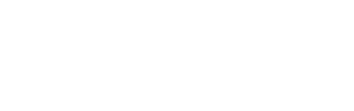
vector<string> arrсreate

1.2. Поиск слов-анаграмм

vector<string>



A



vector<string> a



Входные данные



Результат

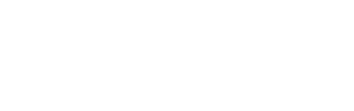


vector<string> checkingallarray

1.3. Вывод вектора со словами-анаграммами



A



vector<string> &x



Входные данные



Результат



void outputvector

## 2. Декомпозиция

* 1. Задачу следует разбить на следующие подзадачи:

1. Создание вектора слов из строки
2. Проверка каждого слова с помощью цикла с сортировкой
3. Вывод слов-анаграмм в консоль

## 3. Реализация функций

vector<string> arrсreate(string str) {

vector<string> temp;

string s = "";

int k = 0;

for (int i = 0; i < str.length(); i++) {

if (str[i] != ' ') {

s = s + str[i];

} else {

if (s != "") {

temp.push\_back(s);

s = "";

}

} }

temp.push\_back(s);

return temp;

}

void outputvector(vector<string> &x) {

for (auto const &element: x) {

cout << element << ' ';

}

cout << endl;

}

vector<string> checkingallarray(vector<string> a) {

vector<string> temp;

for (int i = 0; i < a.size(); i++) { // проход по элементам массива

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

string temp1 = a[i];

string temp2 = a[j];

sort(temp1.begin(), temp1.end(), greater<>());

sort(temp2.begin(), temp2.end(), greater<>());

if (temp1 == temp2 && i != j) {

temp.push\_back(a[i]);

}

}

}

return temp;

}

## 4. Кодирование алгоритма программы

#include <iostream>

#include <string>

#include <algorithm>

#include <vector>

using namespace std;

int main() {

setlocale(LC\_ALL, "rus");;

string str;

getline(cin, str);

vector<string> z = arrсreate(str);

vector<string> result = checkingallarray(z);

outputvector(result);

return 0;

## 5. Таблица тестов программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Исходные данные | Ожидаемый результат | Результат программы | Тест пройден/не пройден |
| 1 | cat tac ssa ddd | cat tac | cat tac | Тест пройден |
| 2 | man www nam | man nam | man nam | Тест пройден |

# Вывод

При выполнении практической работы были освоены навыки обработки null терминальных строк и строк класса String. Были разработаны программные решения задач, предложенных в соответствующем варианте.

# Список информационных источников

* Учебник по C++ [http://www.lmpt.univ-tours.fr/~volkov/C++.pdf](https://d.docs.live.net/793bf2b94bcfd004/Рабочий%20стол/http:/www.lmpt.univ-tours.fr/~volkov/C++.pdf)
* Документация по языку C++ [https://docs.microsoft.com/ruru/cpp/?view=msvc-160](https://d.docs.live.net/793bf2b94bcfd004/Рабочий%20стол/https:/docs.microsoft.com/ruru/cpp/?view=msvc-160)