**Министерство науки и высшего образования РФ**

**Пензенский государственный университет**

**Кафедра «Вычислительная техника»**

**ОТЧЁТ**

По лабораторной работе №1

По дисциплине «Логика и основы алгоритмизации в инженерных задачах»

На тему «Простые структуры данных»

**Выполнил студент группы 24ВВВ2:**

Лисицын Н.С.

**Приняли:**

Юрова О.В.

Деев М.В.

**Пенза, 2025**

**Цель работы**

Исследование методов работы со статическими одномерными и двумерными массивами. Закрепление навыков обработки массивов: заполнения, поиска, сортировки и анализа данных.

**Лабораторное задание**

* написать программу, вычисляющую разницу между максимальным и минимальным элементами массива.
* написать программу, реализующую инициализацию массива случайными числами.
* написать программу, реализующую создание массива произвольного размера, вводимого с клавиатуры.
* написать программу, вычисляющую сумму значений в каждом столбце (или строке) двумерного массива.
* написать программу, осуществляющую поиск среди структур student структуру с заданными параметрами (фамилией, именем и т.д.).

**Описание метода решения задачи**

Добавляются необходимые библиотеки и в функции начала программы просим пользователя ввести цифру в консоли для выбора функции.

Дальше реализуем функции:

* «findMaxMinDiff» находит разницу между максимальным и минимальным элементом массива.
* «randomInitArray» заполняет массив случайными числами.
* «createDynamicArray» создаёт массив произвольного размера.
* «sumRowsColsMatrix» суммирует строки и столбцы матрицы с динамической памятью.
* «searchStudentByName» ищет студента по фамилии.

**Листинг**

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#include <locale.h>

typedef struct {

char familia[20];

char imya[20];

char fakultet[20];

int zach\_num;

} Student;

void findMaxMinDiff();

void randomInitArray();

void createDynamicArray();

void sumRowsColsMatrix(int rows, int cols);

void searchStudentByName();

int main() {

setlocale(LC\_ALL, "Russian");

int choice;

do {

printf("\nМеню выбора действий:\n"

"1. Найти разницу между максимумом и минимумом массива\n"

"2. Инициализировать массив случайными числами\n"

"3. Создать массив произвольного размера\n"

"4. Суммировать строки и столбцы матрицы\n"

"5. Искать студента по имени\n"

"0. Выход\n"

"\nВаш выбор: ");

scanf\_s("%d", &choice);

switch (choice) {

case 1:

findMaxMinDiff(); break;

case 2:

randomInitArray(); break;

case 3:

createDynamicArray(); break;

case 4:

sumRowsColsMatrix(3, 3); break;

case 5:

searchStudentByName(); break;

case 0:

printf("Завершение работы...\n");

break;

default:

printf("Некорректный выбор. Повторите попытку.\n");

}

} while (choice != 0);

return 0;

}

// Функция нахождения разницы между максимальным и минимальным элементом массива 1

void findMaxMinDiff() {

const int N = 5;

int arr[N] = { 10, 20, 30, 40, 50 };

int max = arr[0], min = arr[0], diff;

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

if (arr[i] > max) max = arr[i];

else if (arr[i] < min) min = arr[i];

}

diff = max - min;

printf("Максимальное значение: %d\nМинимальное значение: %d\nРазница: %d\n", max, min, diff);

}

// Функция заполнения массива случайными числами 2

void randomInitArray() {

const int N = 10;

int arr[N];

srand(time(NULL));

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

arr[i] = rand() % 100 + 1; // Числа от 1 до 100

}

printf("Случайный массив: ");

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

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

}

printf("\n");

}

// Функция создания массива произвольного размера 3

void createDynamicArray() {

int\* arr;

int size, i;

printf("Введите размер массива: ");

scanf\_s("%d", &size);

arr = (int\*)malloc(size \* sizeof(int));

if (arr == NULL) {

printf("Ошибка выделения памяти!\n");

return;

}

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

printf("Введите элемент #%d: ", i + 1);

scanf\_s("%d", &arr[i]);

}

printf("\nЗаполненный массив: ");

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

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

}

printf("\n");

free(arr);

}

// Функция суммирования строк и столбцов матрицы с динамической памятью 4

void sumRowsColsMatrix(int rows, int cols) {

int\*\* mat = (int\*\*)malloc(rows \* sizeof(int\*));

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

mat[i] = (int\*)malloc(cols \* sizeof(int));

}

int value = 1;

for (int r = 0; r < rows; r++) {

for (int c = 0; c < cols; c++) {

mat[r][c] = value++;

}

}

int\* rowSums = (int\*)calloc(rows, sizeof(int));

int\* colSums = (int\*)calloc(cols, sizeof(int));

for (int r = 0; r < rows; r++) {

for (int c = 0; c < cols; c++) {

rowSums[r] += mat[r][c];

}

}

for (int c = 0; c < cols; c++) {

for (int r = 0; r < rows; r++) {

colSums[c] += mat[r][c];

}

}

printf("Матрица (%dx%d):\n", rows, cols);

for (int r = 0; r < rows; r++) {

for (int c = 0; c < cols; c++) {

printf("%d\t", mat[r][c]);

}

printf("\n");

}

printf("\nСуммы строк: ");

for (int r = 0; r < rows; r++) {

printf("%d ", rowSums[r]);

}

printf("\nСуммы столбцов: ");

for (int c = 0; c < cols; c++) {

printf("%d ", colSums[c]);

}

printf("\n");

for (int i = 0; i < rows; ++i) free(mat[i]); // Освобождаем каждую строку

free(mat); // Освобождаем сам массив указателей

free(rowSums); // Освобождаем сумму строк

free(colSums); // Освобождаем сумму столбцов

}

// Функция поиска студента по фамилии 5

void searchStudentByName() {

Student students[] = {

{"Ivanov", "Иван", "Инф.", 12345},

{"Petrov", "Петр", "Мат.", 54321},

{"Sidorov", "Сергей", "Физ.", 67890}

};

char searchFamilia[20];

int found = 0;

printf("Введите фамилию студента для поиска: ");

scanf\_s("%s", searchFamilia, sizeof(searchFamilia));

for (int i = 0; i < 3; i++) {

if (strcmp(students[i].familia, searchFamilia) == 0) {

printf("\nСтудент найден:\n"

"Фамилия: %s\nИмя: %s\nФакультет: %s\nЗачётная книжка: %d\n",

students[i].familia, students[i].imya,

students[i].fakultet, students[i].zach\_num);

found = 1;

break;

}

}

if (!found) {

printf("Студент с указанной фамилией не найден.\n");

}

}

**Результат работы программы**

**Изображение выглядит как текст, снимок экрана, Шрифт, алгебра

Содержимое, созданное искусственным интеллектом, может быть неверным.**

**Изображение выглядит как текст, Шрифт, снимок экрана, белый

Содержимое, созданное искусственным интеллектом, может быть неверным.**

**Изображение выглядит как текст, снимок экрана, Шрифт, документ

Содержимое, созданное искусственным интеллектом, может быть неверным.**

**Изображение выглядит как текст, снимок экрана, Шрифт, алгебра

Содержимое, созданное искусственным интеллектом, может быть неверным.**

**Изображение выглядит как текст, снимок экрана, Шрифт, документ

Содержимое, созданное искусственным интеллектом, может быть неверным.**

**Вывод**

В ходе данной лабораторной работы были исследованы методы работы со статическими одномерными и двумерными массивами. Закреплены навыки обработки массивов: заполнения, поиска, сортировки и анализа данных.

**GitHub**

Ссылка репозитория на GitHub:

<https://github.com/NikitaFoxze/LogicLabs.git>