**Министерство науки и высшего образования Российской Федерации**

**Пензенский государственный университет**

**Кафедра “Вычислительная техника”**

ОТЧЁТ

По лабораторной работе №1

По курсу «Логика и основы алгоритмизации в инженерных задачах»

На тему «Простые структуры данных»

Вариант №1

Выполнил студент группы 23ВВВ4:

Брагин А.М.

Приняли:

к.т.н. доцент Деев М.В. \_\_\_\_

к.т.н. доцент Юрова О.В. \_\_\_\_

Пенза 2024

**Лабораторное задание**

**Задание 1**: написать программу, вычисляющую разницу между максимальным и минимальным элементами массива.

**Задание 2**: написать программу, реализующую инициализацию массива случайными числами.

**Задание 3**: написать программу, реализующую создание массива произвольного размера, вводимого с клавиатуры.

**Задание 4**: написать программу, вычисляющую сумму значений в каждом столбце (или строке) двумерного массива.

**Задание 5**: написать программу, осуществляющую поиск среди структур student структуру с  заданными параметрами (фамилией, именем и т.д.).

### Листинг

**Задания 1-4)**

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <locale.h>

#define \_CRT\_SECURE\_NO\_WARNINGS

// Задание 1: Вычисление разницы между максимальным и минимальным элементами массива

int findDifference(int\* array, int size) {

int max = array[0];

int min = array[0];

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

if (array[i] > max)

max = array[i];

if (array[i] < min)

min = array[i];

}

return max - min;

}

// Задание 2: Инициализация массива случайными числами

void initializeArray(int\* array, int size) {

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

array[i] = rand() % 101;

}

}

// Функция для запроса размера квадратичной матрицы у пользователя

int getMatrixSize() {

int size;

printf("Введите размер квадратичной матрицы: ");

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

return size;

}

// Функция для заполнения квадратичной матрицы числами от -40 до 20

void initializeMatrix(int\*\* matrix, int size) {

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

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

matrix[i][j] = rand() % 61 - 40; // Генерация чисел от -40 до 20

}

}

}

// Функция для подсчета количества отрицательных элементов ниже главной диагонали

int countNegativeBelowDiagonal(int\*\* matrix, int size) {

int count = 0;

for (int i = 1; i < size; i++) { // Начинаю с 1, чтобы не включать элементы на главной диагонали

for (int j = 0; j < i; j++) { // Иду только по элементам ниже главной диагонали

if (matrix[i][j] < 0) {

count++;

}

}

}

return count;

}

void sumRows(int\*\* array, int rows, int cols) {

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

int sum = 0;

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

sum += array[i][j];

}

printf("Сумма строки %d: %d\n", i + 1, sum);

}

}

int main() {

setlocale(LC\_ALL, "Russian");

srand(time(NULL));

// Задание 3: Создание массива произвольного размера

int size;

printf("Введите размер одномерного массива: ");

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

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

// Инициализация и вывод одномерного массива

initializeArray(array, size);

printf("Одномерный массив: ");

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

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

}

printf("\n");

// Вычисление разницы между максимальным и минимальным элементами

int difference = findDifference(array, size);

printf("Разница между максимальным и минимальным элементами: %d\n", difference);

// Работа с двумерным массивом

int rows, cols;

printf("Введите количество строк двумерного массива: ");

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

printf("Введите количество столбцов двумерного массива: ");

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

int\*\* array2D = (int\*\*)malloc(rows \* sizeof(int\*));

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

array2D[i] = (int\*)malloc(cols \* sizeof(int));

initializeArray(array2D[i], cols);

}

// Вывод двумерного массива

printf("Двумерный массив:\n");

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

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

printf("%d ", array2D[i][j]);

}

printf("\n");

}

// Вычисление суммы значений в каждой строке двумерного массива

sumRows(array2D, rows, cols);

// Работа с квадратичной матрицей

int matrixSize = getMatrixSize(); // Получаем размер матрицы от пользователя

int\*\* matrix = (int\*\*)malloc(matrixSize \* sizeof(int\*));

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

matrix[i] = (int\*)malloc(matrixSize \* sizeof(int));

}

// Заполнение матрицы и вывод

initializeMatrix(matrix, matrixSize);

printf("Квадратичная матрица:\n");

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

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

printf("%d ", matrix[i][j]);

}

printf("\n");

}

// Подсчёт отрицательных элементов ниже главной диагонали

int negativeCount = countNegativeBelowDiagonal(matrix, matrixSize);

printf("Количество отрицательных элементов ниже главной диагонали: %d\n", negativeCount);

// Освобождение выделенной памяти

free(array);

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

free(array2D[i]);

}

free(array2D);

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

free(matrix[i]);

}

free(matrix);

return 0;

}

**Задание 5)**

**#define \_CRT\_SECURE\_NO\_WARNINGS**

**#include <stdio.h>**

**#include <string.h>**

**#include <stdlib.h>**

**int main(void) {**

**setvbuf(stdin, NULL, \_IONBF, 0);**

**setvbuf(stdout, NULL, \_IONBF, 0);**

**int i;**

**struct student {**

**char famil[20];**

**char name[20];**

**char facult[20];**

**int Nomzach;**

**} stud[3];**

**// Ввод данных о студентах**

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

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

**scanf("%19s", stud[i].famil);**

**}**

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

**printf("imya %s\n", stud[i].famil);**

**scanf("%19s", stud[i].name);**

**}**

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

**printf("fakult %s %s\n", stud[i].famil, stud[i].name);**

**scanf("%19s", stud[i].facult);**

**}**

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

**printf("zachetka %s %s\n", stud[i].famil, stud[i].name);**

**scanf("%d", &stud[i].Nomzach);**

**}**

**// Вывод данных о студентах**

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

**printf("student %s %s obuchaetsa na fak %s, nomer zachetki %d\n",**

**stud[i].famil, stud[i].name, stud[i].facult, stud[i].Nomzach);**

**}**

**// Поиск студента**

**char search\_param[20];**

**printf("Vvedite parametrl dlya poiska (familiya, imya, nomer zachetki): ");**

**scanf("%19s", search\_param);**

**int found = 0; // Флаг для проверки, найден ли студент**

**// Проверяем, является ли введённое значение числом (для поиска по номеру зачётной книжки)**

**char\* endptr;**

**long int search\_nomzach = strtol(search\_param, &endptr, 10);**

**// Если это число, ищем по номеру зачётной книжки**

**if (\*endptr == '\0') {**

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

**if (stud[i].Nomzach == search\_nomzach) {**

**printf("naiden student: %s %s, fak: %s, zach: %d\n",**

**stud[i].famil, stud[i].name, stud[i].facult, stud[i].Nomzach);**

**found = 1;**

**break;**

**}**

**}**

**}**

**else { // Иначе ищем по фамилии или имени**

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

**if (strcmp(stud[i].famil, search\_param) == 0 || strcmp(stud[i].name, search\_param) == 0) {**

**printf("naiden student: %s %s, fak: %s, zach: %d\n",**

**stud[i].famil, stud[i].name, stud[i].facult, stud[i].Nomzach);**

**found = 1;**

**break;**

**}**

**}**

**}**

**if (!found) {**

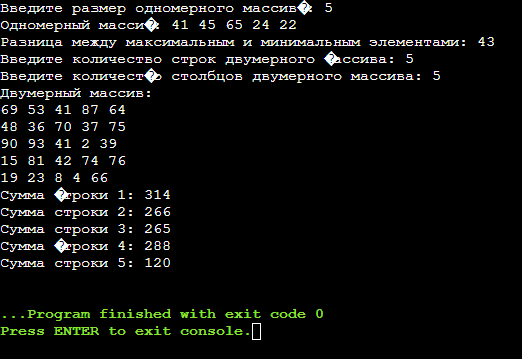
**printf("student s parametrom %s ne naiden\n", search\_param);**

**}**

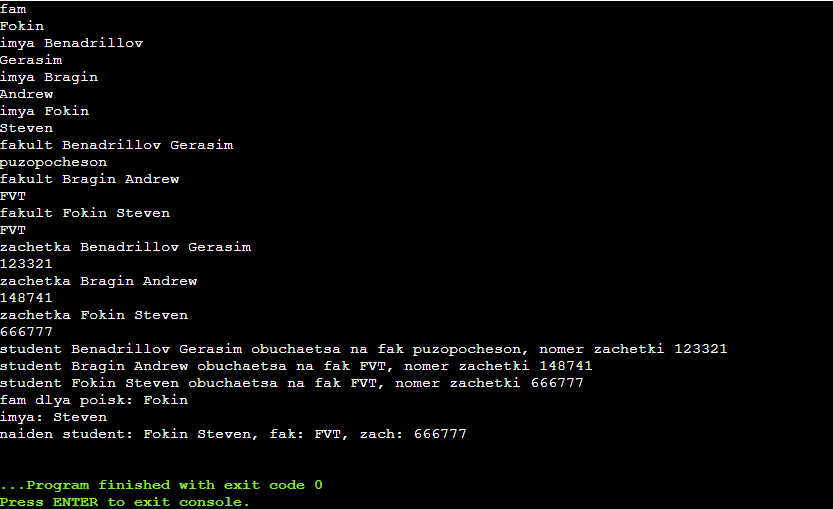
**return 0;**

**}**

**Результат работы программы**



**Рисунок 1 — Результаты работы программы с заданиями 1-4.**



**Рисунок 2 — Результаты работы программы с заданиями 5.**