**ОТЧЕТ**

по лабораторной работе №1

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Простые структуры данных»

Выполнили:

студенты группы 23ВВВ4

Гришаев. Д.А.

Фельдман Г.О.

Приняли:

Деев М.В.

Юрова О.В.

**Название**

Простые структуры данных

**Цель работы**

Повторение принципов использования и применения простых структур данных языка Си (массивов, строк) для решения поставленных задач

**Лабораторное задание**

На основе приведенных в лабораторной работе примеров выполнить следующие задания:

1. **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.

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

**Описание метода решения задачи**

а) Задаем массив произвольного размера, вводимого с клавиатуры. Далее выделяем память под указатель и заполняем его же рандомными числами в некотором диапазоне. В конечном итоге сравниваем с заданным максимумом и минимумом заполненные числами отбираем нужные конечные значения.

Для задания №4 создаем двумерный массив, заполняем отдельно первый столбец и второй столбец числами и складываем на каждой итерации, это и будет необходимая сумма строки.

b) Создаем структуру, включающую фамилию, имя, факультет, а также номер зачетной книжки студента. Далее осуществляем варианты по которому будет производиться поиск. Введенные в последствии данные пользователя, будут сравниваться с введенной изначально базой на поиск соответствий.

### Листинг

а)

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int main() {

srand(time(NULL));

int size\_mass, max = -1000, min = 1000;

printf("enter array z\size: ");

scanf("%i", &size\_mass);

int\* mass = (int\*)malloc(sizeof(int) \* size\_mass);

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

\*mass = rand() % 90 - 40;

if (\*mass > max) max = \*mass;

if (\*mass < min) min = \*mass;

printf(" %i,", \*mass);

}

printf("\nmax: %i min: %i, diff = %i\n", max, min, max - min);

int a[2][10], otv;

for (int i = 0; i < 10; i++) {

a[0][i] = rand() % 100;

a[1][i] = rand() % 100;

otv = a[0][i] + a[1][i];

printf("%i, %i, sum = %i\n", a[0][i], a[1][i], otv);

}

free(mass);

}

b)

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <ctype.h>

#include <string.h>

char\* tolowerstr(char str[20]) {

char ret[20], i;

for (i = 0; str[i]; i++) {

ret[i] = tolower(str[i]);

}

ret[i] = '\0';

return(ret);

};

int main(void) {

system("chcp 1251");

char i;

struct student {

char famil[20], name[20], facult[20];

int Nomzach;

} stud[3] = {

{"Felman","Bleb","OFT",1337},

{"Gigaev","Mima","MVD",2020},

{"Kiryaev","Kurill","ZOV",2048}

};

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

printf("Cтудент %s %s обучается на факультете %s, номер зачётной книжки %d \n", stud[i].famil, stud[i].name,stud[i].facult, stud[i].Nomzach);

}

printf("\nВыберите вариант поиска:\n1 - поиск по фамилии\n2 - поиск по имени\n3 - поиск по факультету\n4 - поиск по номеру зачетной книжки\n");

char mode = \_getch();

switch (mode) {

case '1':{

printf("\nВведите фамилию: ");

char search[20];

scanf("%20s", &search);

strcpy(search,tolowerstr(search));

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

if (strcmp(tolowerstr(stud[i].famil), search) == 0) {

printf("Cтудент %s %s обучается на факультете %s, номер зачётной книжки %d \n", stud[i].famil, stud[i].name, stud[i].facult, stud[i].Nomzach);

}

}

break;

}

case '2':{

printf("\nВведите имя: ");

char search[20];

scanf("%20s", &search);

strcpy(search, tolowerstr(search));

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

if (strcmp(tolowerstr(stud[i].name), search) == 0) {

printf("Cтудент %s %s обучается на факультете %s, номер зачётной книжки %d \n", stud[i].famil, stud[i].name, stud[i].facult, stud[i].Nomzach);

}

}

break;

}

case '3':{

printf("\nВведите факультет: ");

char search[20];

scanf("%20s", &search);

strcpy(search, tolowerstr(search));

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

if (strcmp(tolowerstr(stud[i].facult), search) == 0) {

printf("Cтудент %s %s обучается на факультете %s, номер зачётной книжки %d \n", stud[i].famil, stud[i].name, stud[i].facult, stud[i].Nomzach);

}

}

break;

}

case '4':

default: {

printf("\nВведите номер зачетной книжки: ");

int search;

scanf("%i", &search);

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

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

printf("Cтудент %s %s обучается на факультете %s, номер зачётной книжки %d \n", stud[i].famil, stud[i].name, stud[i].facult, stud[i].Nomzach);

}

}

break;

}

}

}

**Пояснительный текст к программе**

a) rand() будет возвращать одинаковые последовательности чисел. Поэтому был использована функция srand(), аргументом которой является инициализирующее целое число. Передав в time() в качестве параметра NULL, мы получим целое число, которое можно передать в srand(), установив, таким образом уникальное инициализирующее значение генератора

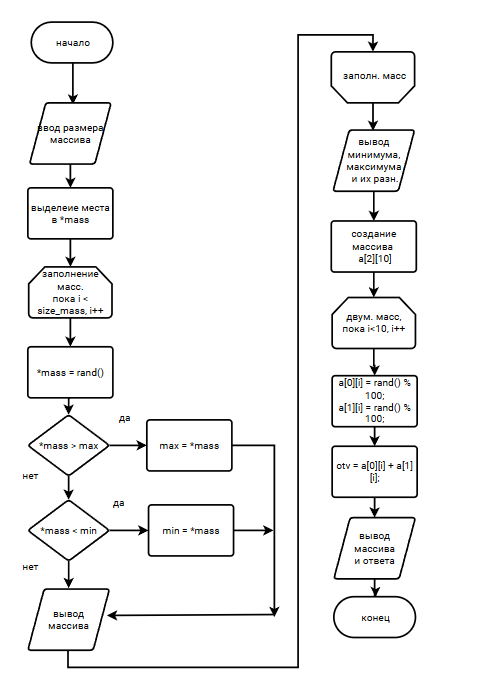
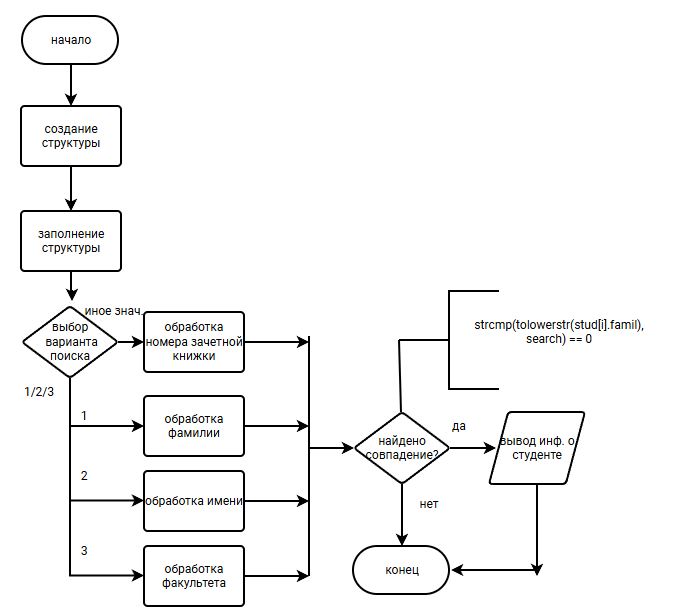
srand(time(NULL));

malloc() - функция динамического выделения памяти.

sizeof() - функция возвращает количество байт, занимаемое элементом указанного типа.

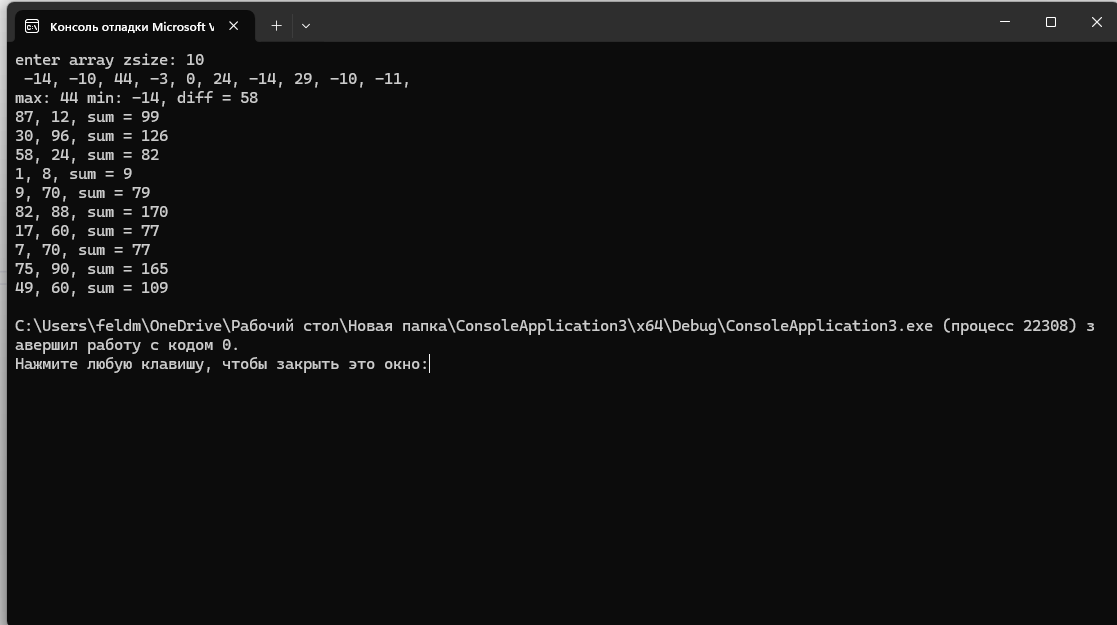
b) tolowerstr() – интерпретирует заглавные буквы в строчные

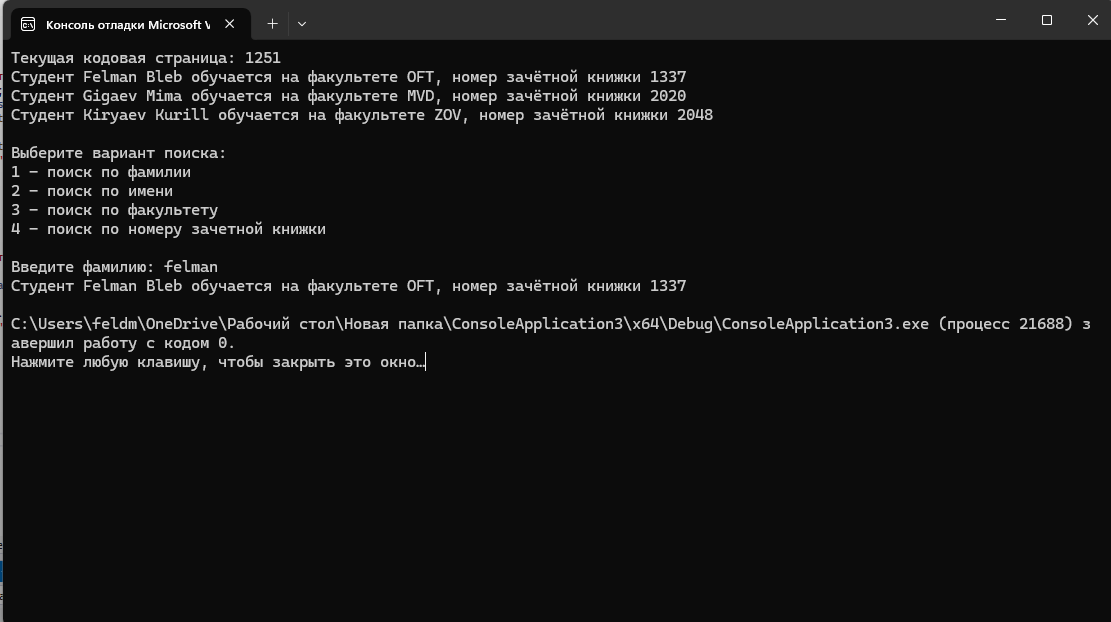
**Схема программы**

**  
 Рисунок 1 — Схема программы а.**

**Рисунок 2 — Схема программы б.**

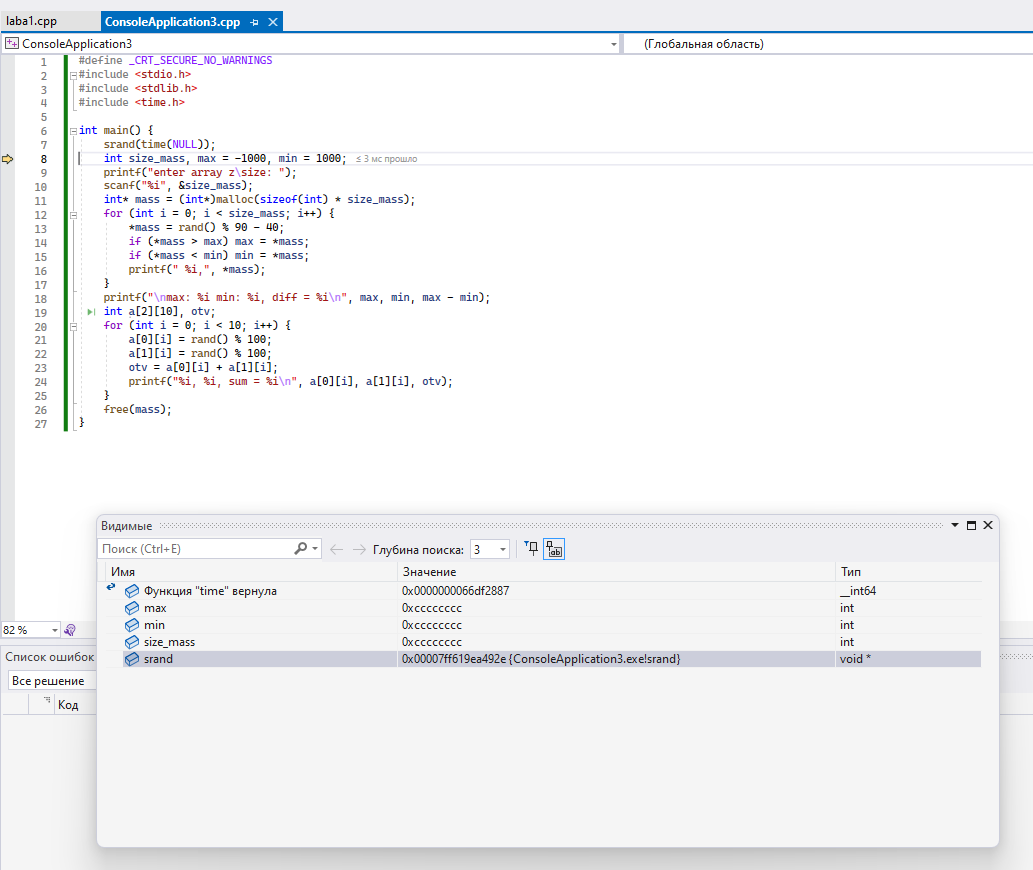
**Результат работы программы**

 **Рисунок 3 — Результаты работы программы а.**

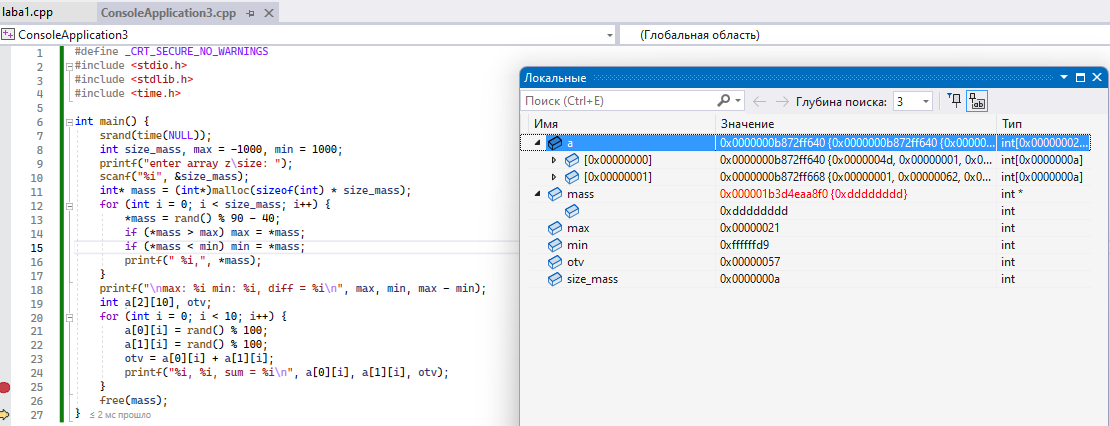
 **Рисунок 4 — Результаты работы программы b.**

### Протокол трассировки программы

На рисунке 5 показан протокол трассировки до заполнения массива.

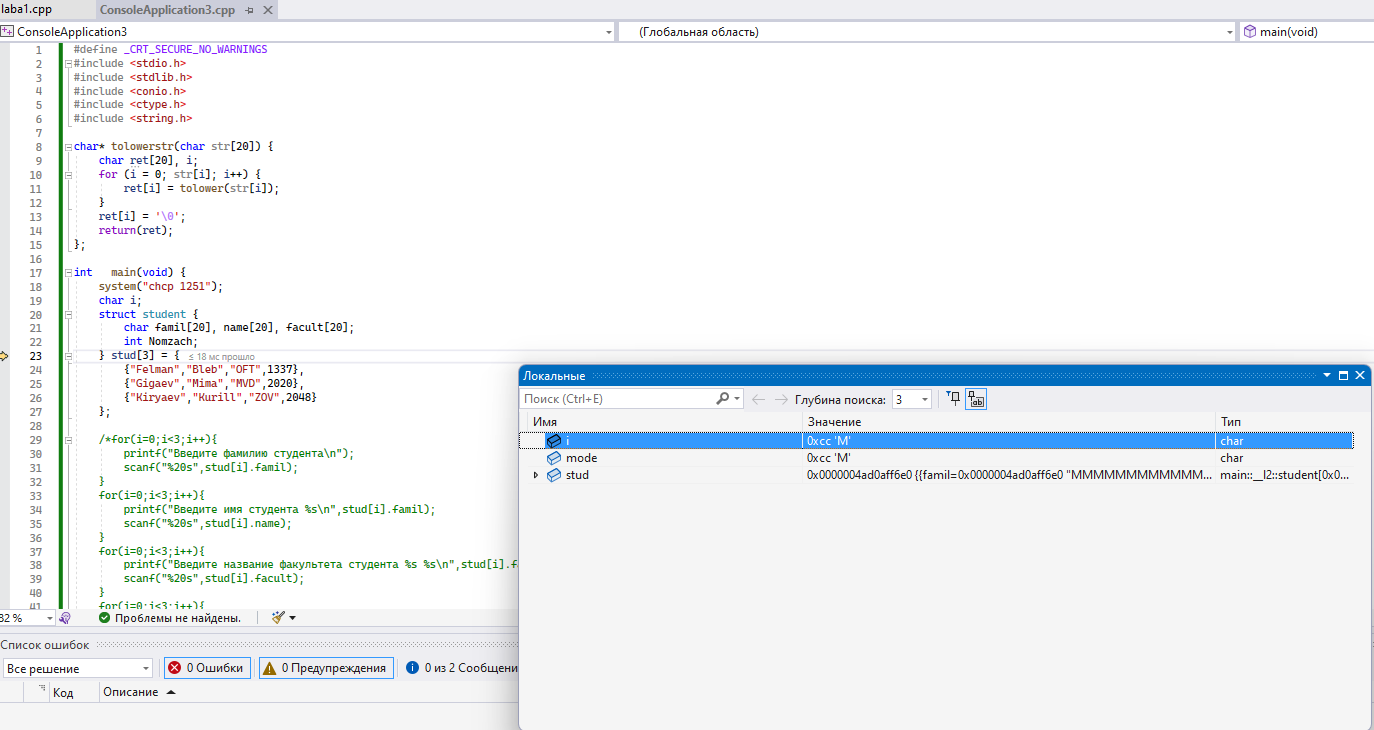
  
**Рисунок 5 — Протокол трассировки прогр. а до зап.**

На рисунке 6 показан протокол трассировки после заполнения массива

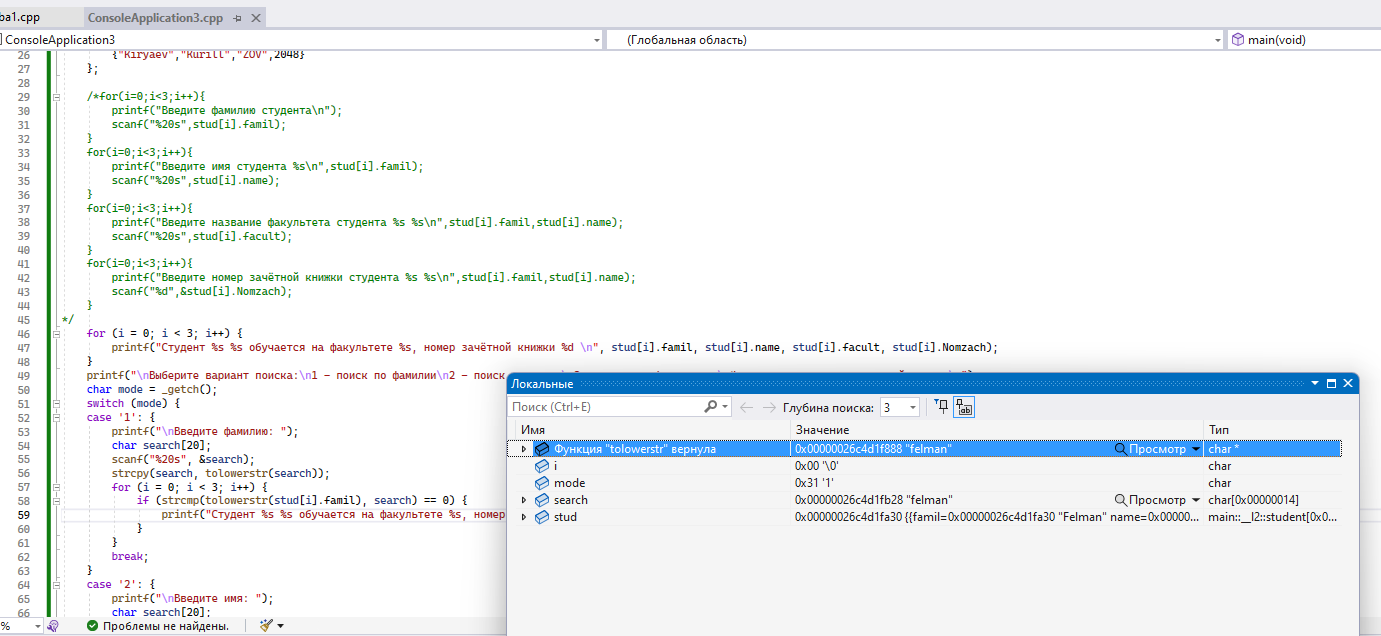


**Рисунок 6 — Протокол трассировки прогр. а после зап.**

На рисунке 7 показан протокол трассировки до заполнения структур.

  
**Рисунок 7 — Протокол трассировки программы b до зап.**

На рисунке 8 показан протокол трассировки после заполнения структур и поиска.

  
**Рисунок 8 — Протокол трассировки b после зап.**

Результат работы программы, показанный на рисунке 1, совпал с результатами трассировки.

### Выводы

Повторили принципы использования и применения простых структур данных языка Си (массивов, строк) для решения поставленных задач.