Министерство образования Российской Федерации

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 студенты группы 19ВВ3:**

Ерёмин А.А

**Принял:**

Митрохин М.А

Пенза 2020

**Цель:** изучить простые структуры данных.

**Ход работы:**

Для удобства задания 1-4 были выполнены в одном файле.

**Задание 1:** написать программу, вычисляющую разницу между максимальным и

минимальным элементами массива.

Чтобы вычислить максимальный и минимальный элемент массива, воспользуемся простым алгоритмом сравнения. Для этого присвоим переменной Max значение первого элемента массива и будем сравнить его с другими элементами, попутно присваивая Max значение больших элементов. Такая же суть применяется и для поиска минимального значения. Дальше просто вычисляем модуль разницы Max и Min. Данный алгоритм реализован в функции SearchMaxMin() представленной ниже.

void SearchMaxMin(int\* Numbers, int Size) {

int Max, Min;

Max = Numbers[0]; // Присваиваем первый элемент массива

Min = Numbers[0];

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

if (Max < Numbers[i + 1]) { // Сравниваем элементы и присваиваем наибольшее из 2-ух

Max = Numbers[i + 1];

}

if (Min > Numbers[i + 1]) { // Сравниваем элементы и присваиваем наименьшее из 2-ух

Min = Numbers[i + 1];

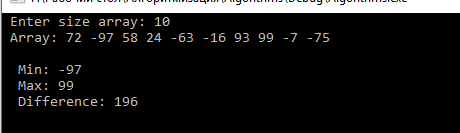
}

}

printf("\n Min: %d\n Max: %d", Min, Max);

printf("\n Difference: %d", abs(Max - Min));

}



**Задание 2:** написать программу, реализующую инициализацию массива

случайными числами.

Для этого достаточно просто воспользоваться функцией rand().

void RandomArray(int\* Numbers, int Size) {

printf(" Array: ");

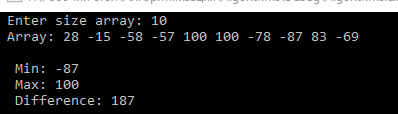
for (int i = 0; i < Size; i++) {

Numbers[i] = rand() % 201 - 100; // Заполнение массива случайными цифрами в диапазоне от -100 до 100

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

}

}



**Задание 3:** написать программу, реализующую создание массива произвольного

размера, вводимого с клавиатуры.

Запрашиваем ввод с клавиатуры и выделяем память под массив, не забывая освободить память после использования.

int\* Numbers;

int Size;

printf(" Enter size array: "); // Пользовательский ввод размера массива

scanf("%d", &Size);

Numbers = (int\*)malloc(sizeof(int) \* Size); // Выделение памяти

free(Numbers); // Освобождение памяти



**Задание 4:** написать программу, вычисляющую сумму значений в каждом столбце

(или строке) двумерного массива.

int Matrix[ROWS][COLS];

int SumCols = 0;

printf("\n\n");

RandomMatrix(Matrix); // Инициализвация массива случайными числами

printf("\n\n");

for (int j = 0; j < COLS; j++) { // Т.к считается сумма столбца, меняем циклы местами

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

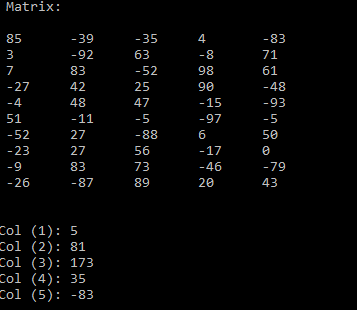
SumCols += Matrix[i][j];

}

printf("Col (%d): %d\n", j + 1, SumCols);

SumCols = 0; // Обнуляем для считывания следующих столбцов

}



**Задание 5:** написать программу, осуществляющую поиск среди структур student

структуру с заданными параметрами (фамилией, именем и т.д.).

typedef struct students {

char LastName[20];

char Name[20];

int NumberBook; // Номер зачётной книжки

int Number;

};

void SearchStruct(students students[]) {

char SearchName[20];

printf("Enter name to search: ");

scanf("%s", SearchName);

bool flag = false;

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

if (strcmp(SearchName, students[i].Name) == 0) { // Поиск в структурах по имени

flag = true;

printf("Search success. Number struct is: %d\n", students[i].Number);

printf(" Name: %s\n Last name: %s\n Number book: %d", students[i].LastName, students[i].Name, students[i].NumberBook);

}

}

if (!flag) {

printf("Search failed.");

}

}

int main() {

students students[SIZE];

for (int i = 0; i < SIZE; i++) { // Инициализируем структуры

printf("Enter name: ");

scanf("%s", students[i].Name);

printf("Enter last name: ");

scanf("%s", students[i].LastName);

printf("Enter number book: ");

scanf("%d", &students[i].NumberBook);

students[i].Number = i + 1;

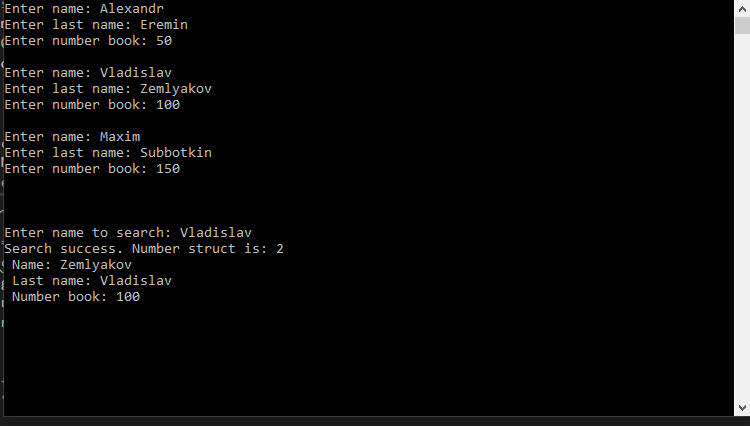
printf("\n");

}

printf("\n\n");

SearchStruct(students);

}



**Листинг.**

Полный листинг программ:

**Задания 1-4:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "Algorithms.h"

#define ROWS 10 // Константы для матрицы

#define COLS 5

void SearchMaxMin(int\* Numbers, int Size) {

int Max, Min;

Max = Numbers[0]; // Присваиваем первый элемент массива

Min = Numbers[0];

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

if (Max < Numbers[i + 1]) { // Сравниваем элементы и присваиваем наибольшее из 2-ух

Max = Numbers[i + 1];

}

if (Min > Numbers[i + 1]) { // Сравниваем элементы и присваиваем наименьшее из 2-ух

Min = Numbers[i + 1];

}

}

printf("\n Min: %d\n Max: %d", Min, Max);

printf("\n Difference: %d", abs(Max - Min));

}

void RandomArray(int\* Numbers, int Size) {

printf(" Array: ");

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

Numbers[i] = rand() % 201 - 100; // Заполнение массива случайными цифрами в диапазоне от -100 до 100

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

}

}

void RandomMatrix(int Matrix[ROWS][COLS]) {

printf(" Matrix: \n\n");

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

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

Matrix[i][j] = rand() % 201 - 100; // Заполнение массива случайными цифрами в диапазоне от -100 до 100

printf(" %d\t", Matrix[i][j]);

}

printf("\n");

}

}

int main() {

// Задание 1,2,3

srand(time(NULL));

int\* Numbers;

int Size;

printf(" Enter size array: "); // Пользовательский ввод размера массива

scanf("%d", &Size);

Numbers = (int\*)malloc(sizeof(int) \* Size); // Выделение памяти

RandomArray(Numbers, Size); // Функция по инициализации массива случайными числами

printf("\n");

SearchMaxMin(Numbers, Size); // Поиск максимального и минимального элементов и вывод разницы

free(Numbers);

printf("\n");

// Задание 4

int Matrix[ROWS][COLS];

int SumCols = 0;

printf("\n\n");

RandomMatrix(Matrix); // Инициализвация массива случайными числами

printf("\n\n");

for (int j = 0; j < COLS; j++) { // Т.к считается сумма столбца, меняем циклы местами

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

SumCols += Matrix[i][j];

}

printf("Col (%d): %d\n", j + 1, SumCols);

SumCols = 0; // Обнуляем для считывания следующих столбцов

}

}

**Задание 5:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "Algorithms.h"

#define SIZE 3

typedef struct students {

char LastName[20];

char Name[20];

int NumberBook;

int Number;

};

void SearchStruct(students students[]) {

char SearchName[20];

printf("Enter name to search: ");

scanf("%s", SearchName);

bool flag = false;

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

if (strcmp(SearchName, students[i].Name) == 0) { // Поиск в структурах по имени

flag = true;

printf("Search success. Number struct is: %d\n", students[i].Number);

printf(" Name: %s\n Last name: %s\n Number book: %d", students[i].LastName, students[i].Name, students[i].NumberBook);

}

}

if (!flag) {

printf("Search failed.");

}

}

int main() {

students students[SIZE];

for (int i = 0; i < SIZE; i++) { // Инициализируем структуры

printf("Enter name: ");

scanf("%s", students[i].Name);

printf("Enter last name: ");

scanf("%s", students[i].LastName);

printf("Enter number book: ");

scanf("%d", &students[i].NumberBook);

students[i].Number = i + 1;

printf("\n");

}

printf("\n\n");

SearchStruct(students);

}

**Вывод:** в ходе выполнения лабораторного задания изучил простые структуры данных и выполнил 5 задач.