МИНИСТЕРВСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №1

по курсу «Логика и ОА в ИЗ»

на тему «Простые структуры данных»

Выполнили:

Студенты группы 23ВВВ2

Севостьянов Олег

Кандаков Андрей

Принял:

Митрохин М.А.

Юрова О.В.

Пенза 2024

**Цель работы**

Повторение пройденного материала, связанного с массивами и простыми структурами.

**Лабораторное задание**

Написать программу:

1)Вычисляющую разницу между максимальным и минимальным элементами массива;

2)Реализующую инициализацию массива случайными числами;

3)Реализующую создание массива произвольного размера, вводимого с клавиатуры;

4)Вычисляющую сумму значений в каждом столбце(или строке) двумерного массива;

5)Осуществляющую поиск среди структур student структуру с заданными параметрами (фамилией, именем и т.д.).

**Пояснение к коду**

Задание 1:

- \*\*`int difference\_max\_min(int arr[], int size)`\*\*: Определяет функцию `difference\_max\_min`, которая принимает массив `arr` и его размер `size`, и возвращает целое число.

- \*\*`if (size == 0) { return 0; }`\*\*: Проверяет, если размер массива равен 0 (массив пустой), функция возвращает 0, чтобы избежать ошибок при попытке работы с пустым массивом.

- \*\*`int max\_element = arr[0];`\*\*: Инициализирует переменную `max\_element`, присваивая ей значение первого элемента массива. Она будет использоваться для хранения максимального значения в массиве.

- \*\*`int min\_element = arr[0];`\*\*: Инициализирует переменную `min\_element`, присваивая ей значение первого элемента массива. Она будет использоваться для хранения минимального значения в массиве.

- \*\*`for (int i = 1; i < size; i++)`\*\*: Цикл `for`, который начинает итерирование с 1 до `size-1` (т.е. по всем элементам массива, кроме первого).

- \*\*`if (arr[i] > max\_element) { max\_element = arr[i]; }`\*\*: Проверяет, если текущий элемент массива больше, чем текущий `max\_element`, тогда обновляет `max\_element` значением текущего элемента.

- \*\*`if (arr[i] < min\_element) { min\_element = arr[i]; }`\*\*: Проверяет, если текущий элемент массива меньше, чем текущий `min\_element`, тогда обновляет `min\_element` значением текущего элемента.

- \*\*`return max\_element - min\_element;`\*\*: Возвращает разницу между максимальным и минимальным элементами массива.

- \*\*`int main()`\*\*: Определяет главную функцию программы.

- \*\*`int array[] = { 10, 3, 5, 20, 8 };`\*\*: Создает массив `array` из 5 целых чисел.

- \*\*`int size = sizeof(array) / sizeof(array[0]);`\*\*: Вычисляет размер массива, деля общий размер массива (в байтах) на размер одного элемента.

- \*\*`int result = difference\_max\_min(array, size);`\*\*: Вызывает функцию `difference\_max\_min`, передавая ей массив и его размер, и сохраняет результат в переменной `result`.

- \*\*`printf("Разница %d\n", result);`\*\*: Выводит на экран строку "Разница" и значение переменной `result`.

Задание 2:

- \*\*`int size = 10;`\*\*: Инициализирует переменную `size`, которая задает количество элементов в массиве (здесь — 10).

- \*\*`int\* array = (int\*)malloc(size \* sizeof(int));`\*\*: Динамически выделяет память для массива из 10 целых чисел с помощью функции `malloc`. `size \* sizeof(int)` указывает, сколько байтов памяти нужно выделить (для 10 целых чисел). `int\*` приводит указатель к типу `int\*`.

- \*\*`if (array == NULL) { return 1; }`\*\*: Проверяет, удалось ли выделить память. Если `malloc` вернул `NULL` (нехватка памяти), программа завершится с кодом ошибки 1.

- \*\*`srand(time(NULL));`\*\*: Инициализирует генератор случайных чисел, используя текущее время как начальное значение (считается от 1 января 1970 года). Это гарантирует, что каждый запуск программы будет давать разные случайные числа.

- \*\*`for (int i = 0; i < size; i++)`\*\*: Цикл `for`, который итерируется по каждому элементу массива `array` от 0 до 9 (всего 10 элементов).

- \*\*`array[i] = rand() % 100;`\*\*: Генерирует случайное число с помощью функции `rand()` и берет его остаток от деления на 100 (`% 100`), чтобы получить случайное число в диапазоне от 0 до 99. Это число записывается в текущую позицию массива `array[i]`.

- \*\*`printf("Элемент %d: %d\n", i + 1, array[i]);`\*\*: Выводит на экран индекс текущего элемента массива и его значение. Индекс выводится как `i + 1`, чтобы счет начинался с 1, а не с 0.

Задание 3:

- \*\*`scanf("%d", &size);`\*\*: Считывает введенное пользователем целое число и сохраняет его в переменную `size`.

- \*\*`if (size <= 0)`\*\*: Проверяет, если введенное число меньше или равно 0. Если да, программа выводит сообщение об ошибке и завершает работу.

- \*\*`printf("Размер массива должен быть больше нуля.\n");`\*\*: Выводит сообщение об ошибке, если пользователь ввел неправильный размер массива.

- \*\*`return 1;`\*\*: Прерывает выполнение программы и возвращает код ошибки 1.

- \*\*`int\* array = (int\*)malloc(size \* sizeof(int));`\*\*: Динамически выделяет память для массива целых чисел размером `size`. `malloc` возвращает указатель на выделенную память, который приводится к типу `int\*`. Если памяти недостаточно, `malloc` возвращает `NULL`.

- \*\*`if (array == NULL)`\*\*: Проверяет, удалось ли выделить память. Если указатель равен `NULL`, программа выводит сообщение об ошибке и завершает выполнение.

- \*\*`printf("Ошибка выделения памяти.\n");`\*\*: Выводит сообщение об ошибке, если память не была выделена.

- \*\*`return 1;`\*\*: Завершает выполнение программы с кодом ошибки 1.

- \*\*`free(array);`\*\*: Освобождает выделенную ранее память под массив, что предотвращает утечку памяти.

Задание 4:

- \*\*`int main()`\*\*: Начало главной функции программы.

- \*\*`int rows, cols;`\*\*: Объявление двух переменных `rows` и `cols`, которые будут хранить количество строк и столбцов двумерного массива.

- \*\*`printf`\*\*: Выводит приглашение пользователю ввести количество строк и столбцов.

- \*\*`scanf`\*\*: Считывает введенные пользователем значения и сохраняет их в переменные `rows` и `cols`.

- \*\*`int\*\* array = (int\*\*)malloc(rows \* sizeof(int\*));`\*\*: Выделяет память для массива указателей (каждый элемент — это строка), который будет хранить указатели на строки массива. Количество указателей соответствует числу строк (`rows`).

- \*\*`for (int i = 0; i < rows; i++) { ... }`\*\*: Для каждой строки выделяется память для массива целых чисел (каждый элемент — это столбец). Размер каждой строки равен числу столбцов (`cols`).

- В итоге программа создает двумерный массив `rows x cols`.

- \*\*`printf("Введите элементы массива:\n");`\*\*: Выводит сообщение о начале ввода элементов массива.

- \*\*Два вложенных цикла\*\*:

- \*\*Внешний цикл\*\* (`for (int i = 0; i < rows; i++)`) проходит по строкам массива.

- \*\*Внутренний цикл\*\* (`for (int j = 0; j < cols; j++)`) проходит по столбцам каждой строки.

- \*\*`printf("Элемент [%d][%d]: ", i + 1, j + 1);`\*\*: Выводит приглашение для ввода каждого элемента массива с указанием его индекса.

- \*\*`scanf("%d", &array[i][j]);`\*\*: Считывает введенное значение и сохраняет его в массив.

- \*\*Внешний цикл\*\* (`for (int j = 0; j < cols; j++)`) проходит по каждому столбцу массива.

- \*\*`int sum = 0;`\*\*: Инициализирует переменную `sum`, которая будет хранить сумму элементов текущего столбца.

- \*\*Внутренний цикл\*\* (`for (int i = 0; i < rows; i++)`) проходит по всем строкам данного столбца, складывая значения элементов в переменную `sum`.

- \*\*`printf("Сумма элементов в столбце %d: %d\n", j + 1, sum);`\*\*: После завершения внутреннего цикла выводит сумму элементов для текущего столбца.

- \*\*`for (int i = 0; i < rows; i++)`\*\*: Проходит по каждой строке массива.

- \*\*`free(array[i]);`\*\*: Освобождает память, выделенную для каждой строки массива.

- \*\*`free(array);`\*\*: Освобождает память, выделенную для массива указателей на строки.

Задание 5:

- \*\*`struct student`\*\*: Описание структуры, которая хранит данные о студенте, включая фамилию (`famil`), имя (`name`), факультет (`facult`) и номер зачётной книжки (`Nomzach`).

```c

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

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

```

- Отключает буферизацию ввода и вывода для мгновенного отображения результатов работы программы.

- \*\*`struct student stud[3];`\*\*: Создаётся массив из трёх структур студентов.

- \*\*`int i;`\*\*: Объявляется переменная-счётчик для циклов.

- Цикл для ввода фамилий трёх студентов. Используется функция \*\*`scanf`\*\* для считывания строк длиной до 19 символов (с учётом нулевого символа `\0` в C).

- Объявляются переменные для хранения данных, которые пользователь введёт для поиска: фамилия (\*\*`searchFamil`\*\*), имя (\*\*`searchName`\*\*), факультет (\*\*`searchFacult`\*\*), и номер зачётной книжки (\*\*`searchNomzach`\*\*).

- Переменная \*\*`found`\*\* используется для проверки, найден ли студент (изначально равна 0).

- Программа сравнивает введённые параметры с данными каждого студента. Если совпадения найдены по всем полям, выводятся данные студента, и переменной \*\*`found`\*\* присваивается значение 1. После этого цикл прерывается (`break`).

- Если студент не найден (\*\*`found`\*\* остаётся равным 0), выводится соответствующее сообщение.

**Результат работы программы**

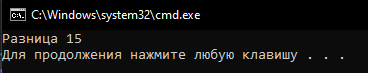
****

Рисунок 1 – результат работы программы (1)

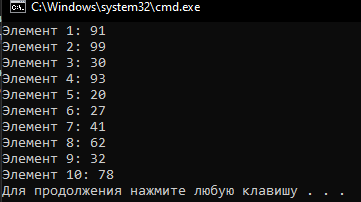
****

Рисунок 2 – результат работы программы (2)

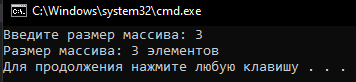
****

Рисунок 3 – результат работы программы (3)

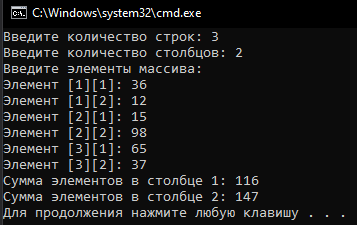
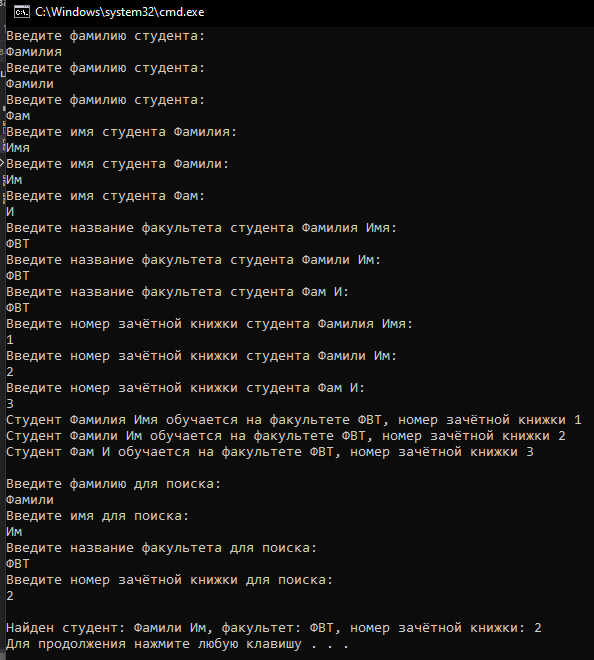
****

Рисунок 4 – результат работы программы (4)

****Рисунок 5 – результат работы программы (5)

**Вывод**

В ходе работы были закреплены знания работы с массивами и простыми структурами.

**Листинг**

**Задание 1**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#include <windows.h>

int difference\_max\_min(int arr[], int size) {

if (size == 0) {

return 0;

}

int max\_element = arr[0];

int min\_element = arr[0];

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

if (arr[i] > max\_element) {

max\_element = arr[i];

}

if (arr[i] < min\_element) {

min\_element = arr[i];

}

}

return max\_element - min\_element;

}

int main() {

int array[] = { 10, 7, 5, 20, 8 };

int size = sizeof(array) / sizeof(array[0]);

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

int result = difference\_max\_min(array, size);

printf("Разница %d\n", result);

return 0;

}

**Задание 2**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#include <windows.h>

#include <time.h>

int main() {

int size = 10;

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

if (array == NULL) {

return 1;

}

srand(time(NULL));

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

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

array[i] = rand() % 100;

printf("Элемент %d: %d\n", i + 1, array[i]);

}

return 0;

}

**Задание 3**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <windows.h>

int main() {

int size;

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

printf("Введите размер массива: ");

scanf("%d", &size);

if (size <= 0) {

printf("Размер массива должен быть больше нуля.\n");

return 1;

}

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

if (array == NULL) {

printf("Ошибка выделения памяти.\n");

return 1;

}

printf("Размер массива: %d элементов\n", size);

free(array);

return 0;

}

**Задание 4**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <windows.h>

int main() {

int rows, cols;

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

printf("Введите количество строк: ");

scanf("%d", &rows);

printf("Введите количество столбцов: ");

scanf("%d", &cols);

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

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

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

}

printf("Введите элементы массива:\n");

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

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

printf("Элемент [%d][%d]: ", i + 1, j + 1);

scanf("%d", &array[i][j]);

}

}

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

int sum = 0;

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

sum += array[i][j];

}

printf("Сумма элементов в столбце %d: %d\n", j + 1, sum);

}

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

free(array[i]);

}

free(array);

return 0;

}

**Задание 5**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <conio.h>

#include <windows.h>

struct student {

char famil[20];

char name[20];

char facult[20];

int Nomzach;

};

int main(void) {

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

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

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

struct student stud[3];

int i;

Ввод данных студентов

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

printf("Введите фамилию студента:\n");

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

}

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

printf("Введите имя студента %s:\n", stud[i].famil);

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

}

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

printf("Введите название факультета студента %s %s:\n", stud[i].famil, stud[i].name);

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

}

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

printf("Введите номер зачётной книжки студента %s %s:\n", stud[i].famil, stud[i].name);

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

}

Вывод данных студентов

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

printf("Студент %s %s обучается на факультете %s, номер зачётной книжки %d\n",

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

}

Поиск по заданным параметрам

char searchFamil[20], searchName[20], searchFacult[20];

int searchNomzach;

int found = 0;

printf("\nВведите фамилию для поиска:\n");

scanf("%19s", searchFamil);

printf("Введите имя для поиска:\n");

scanf("%19s", searchName);

printf("Введите название факультета для поиска:\n");

scanf("%19s", searchFacult);

printf("Введите номер зачётной книжки для поиска:\n");

scanf("%d", &searchNomzach);

Поиск среди студентов

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

if (strcmp(stud[i].famil, searchFamil) == 0 &&

strcmp(stud[i].name, searchName) == 0 &&

strcmp(stud[i].facult, searchFacult) == 0 &&

stud[i].Nomzach == searchNomzach) {

printf("\nНайден студент: %s %s, факультет: %s, номер зачётной книжки: %d\n",

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

found = 1;

break;

}

}

if (!found) {

printf("\nСтудент с такими данными не найден.\n");

}

return 0;

}