Министерство науки и высшего образования РФ

Пензенский государственный университет

Кафедра «Вычислительная техника»

**Отчет**

по лабораторной работе № 1

по курсу «Арифметические и логические основы вычислительной

техники»

на тему «Простые структуры данных»

Выполнил

студент группы 24ВВВ2:

Воеводин И.А.

Гуляевский Д.С.

Каташов К.А.

Приняли

Юрова О.В.

Деев М.В.

Пенза, 2025

**Цель работы**

Освоить методы построения и использования **с**труктур (struct) для организации разнородных данных в едином элементе, а также реализовать поиск по структуре с заданными параметрами.

**Лабораторное задание**

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

**Описание метода решения**

В ходе написания кода для выполнения работы было принято разбить все на несколько файлов. В первом файле выполняется код для задний 1-3 с реализацией алгоритма быстрой сортировки (Quick Sort). Суть алгоритма заключается в том, что он выбирает опорный элемент и перераспределяет другие элементы так, чтобы меньшие оказались слева, а большие — справа от него, затем рекурсивно применяется к получившимся подмассивам.

Во втором файле выполняется код для здания 4, который реализован методом последовательного перебора элементов матрицы для вычисления сумм по столбцам. Создается статический двумерный массив (матрица) размером 3x4 с заранее заданными значениями. Объявляется массив columnSums для хранения сумм каждого столбца. Во вложенных циклах программа проходит по каждому столбцу (j) и для каждого столбца проходит по всем его строкам (i), суммируя элементы. На экран выводится исходная матрица и получившиеся суммы для каждого столбца.

В 3 файле реализован код для 5 задания, в котором присутствуют структуры и линейный список. Для поиска реализованы четыре функции (например, findBySurname), каждая из которых принимает критерий поиска. Алгоритм линейного поиска последовательно перебирает все элементы вектора и сравнивает выбранное поле (например, surname) с заданным значением. Все подходящие записи сохраняются в отдельный вектор, который затем возвращается как результат поиска. Пользовательский интерфейс реализован через консольное меню.

**Листинг**

Задание 1-3

#include <iostream>

#include <time.h>

using namespace std;

void print\_mass(int\* mass, int size);

void quick\_sort(int mass[], int L, int H);

void print\_mass(int\* mass, int size);

void sum\_max\_min(int\* mass, int size);

int main(){

srand(time(0));

int size=0;

cin >> size;

int\* mass = new int[size];

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

mass[i] = rand() % 100-20;

}

cin.ignore();

quick\_sort(mass, 0, size - 1);

print\_mass(mass, size);

sum\_max\_min(mass, size);

delete []mass;

}

void print\_mass(int\* mass, int size) {

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

cout << mass[i]<<endl;

}

}

void sum\_max\_min(int\* mass,int size) {

int sum = mass[0] + mass[size-1];

cout << sum << endl;

}

int partitaion(int mass[], int L, int H) {

int firstHigh = L;

int p = H;

for (int i = L; i < H; ++i) {

if (mass[i] < mass[p]) {

int temp = mass[firstHigh];

mass[firstHigh] = mass[i];

mass[i] = temp;

firstHigh++;

}

}

int temp = mass[firstHigh];

mass[firstHigh] = mass[p];

mass[p] = temp;

return firstHigh;

}

void quick\_sort(int mass[], int L, int H) {

if (H - L > 0) {

int p = partitaion(mass, L, H);

quick\_sort(mass, L, p - 1);

quick\_sort(mass, p + 1, H);

}

}

Задание 4

#include <iostream>

using namespace std;

int main() {

const int ROWS = 3;

const int COLS = 4;

int mass[ROWS][COLS] = {

{1, 2, 3, 4},

{5, 6, 7, 8},

{9, 10, 11, 12}

};

// Вычисление суммы по столбцам

int columnSums[COLS] = { 0 };

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

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

columnSums[j] += mass[i][j];

}

}

// Вывод результатов

cout << "Arrays:" << endl;

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

cout << "\n";

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

cout << mass[i][j] << " ";

}

cout << endl;

}

cout << "\nSum col:" << endl;

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

cout << "col " << j + 1 << ": " << columnSums[j] << endl;

}

return 0;

}

Задание 5

#include <iostream>

#include <string>

#include <windows.h>

#include <vector>

#include <algorithm>

using namespace std;

struct Student {

string surname;

string name;

int age;

string group;

};

// Функция для ввода данных студента

Student inputStudent() {

Student s;

cout << "Фамилия: ";

cin >> s.surname;

cout << "Имя: ";

cin >> s.name;

cout << "Возраст: ";

cin >> s.age;

cout << "Группа: ";

cin >> s.group;

return s;

}

// Функция для вывода данных студента

void printStudent(const Student& s) {

cout << "Фамилия: " << s.surname << endl;

cout << "Имя: " << s.name << endl;

cout << "Возраст: " << s.age << endl;

cout << "Группа: " << s.group << endl;

cout << "------------------------" << endl;

}

// Функции поиска

vector<Student> findBySurname(const vector<Student>& students, const string& surname) {

vector<Student> result;

for (const auto& s : students) {

if (s.surname == surname) {

result.push\_back(s);

}

}

return result;

}

vector<Student> findByName(const vector<Student>& students, const string& name) {

vector<Student> result;

for (const auto& s : students) {

if (s.name == name) {

result.push\_back(s);

}

}

return result;

}

vector<Student> findByAge(const vector<Student>& students, int age) {

vector<Student> result;

for (const auto& s : students) {

if (s.age == age) {

result.push\_back(s);

}

}

return result;

}

vector<Student> findByGroup(const vector<Student>& students, const string& group) {

vector<Student> result;

for (const auto& s : students) {

if (s.group == group) {

result.push\_back(s);

}

}

return result;

}

int main() {

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

vector<Student> students = {

{"Иванов", "Иван", 20, "ИТ-101"},

{"Петров", "Петр", 21, "ИТ-101"},

{"Сидорова", "Мария", 19, "ИТ-102"},

{"Иванов", "Алексей", 20, "ИТ-102"},

{"Кузнецова", "Елена", 22, "ИТ-101"}

};

int choice;

do {

cout << "\n=== ПОИСК СТУДЕНТОВ ===" << endl;

cout << "1. По фамилии" << endl;

cout << "2. По имени" << endl;

cout << "3. По возрасту" << endl;

cout << "4. По группе" << endl;

cout << "5. Показать всех студентов" << endl;

cout << "0. Выход" << endl;

cout << "Выберите вариант: ";

cin >> choice;

switch (choice) {

case 1: {

string surname;

cout << "Введите фамилию: ";

cin >> surname;

auto result = findBySurname(students, surname);

if (result.empty()) {

cout << "Студенты с фамилией '" << surname << "' не найдены." << endl;

}

else {

cout << "Найдено " << result.size() << " студентов:" << endl;

for (const auto& s : result) {

printStudent(s);

}

}

break;

}

case 2: {

string name;

cout << "Введите имя: ";

cin >> name;

auto result = findByName(students, name);

if (result.empty()) {

cout << "Студенты с именем '" << name << "' не найдены." << endl;

}

else {

cout << "Найдено " << result.size() << " студентов:" << endl;

for (const auto& s : result) {

printStudent(s);

}

}

break;

}

case 3: {

int age;

cout << "Введите возраст: ";

cin >> age;

auto result = findByAge(students, age);

if (result.empty()) {

cout << "Студенты возраста " << age << " не найдены." << endl;

}

else {

cout << "Найдено " << result.size() << " студентов:" << endl;

for (const auto& s : result) {

printStudent(s);

}

}

break;

}

case 4: {

string group;

cout << "Введите группу: ";

cin >> group;

auto result = findByGroup(students, group);

if (result.empty()) {

cout << "Студенты группы '" << group << "' не найдены." << endl;

}

else {

cout << "Найдено " << result.size() << " студентов:" << endl;

for (const auto& s : result) {

printStudent(s);

}

}

break;

}

case 5: {

cout << "Все студенты (" << students.size() << "):" << endl;

for (const auto& s : students) {

printStudent(s);

}

break;

}

case 0:

cout << "Выход из программы." << endl;

break;

default:

cout << "Неверный выбор!" << endl;

}

} while (choice != 0);

return 0;

}

**Вывод**

В ходе выполнения лабораторной работы мы вспомнили как работать с одномерными и двумерными массивами, со структурами. Также в ходе написания кода были улучшены знания по таким темам программирования на языке C++ как связанные списки и динамическая память.