Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики

и радиоэлектроники»

Специальность «Программная инженерия»

Учебная дисциплина «Основы алгоритмизации и программирования»

Отчёт

по лабораторной работе No8

*«Функции*»

Подготовил: Студент

гр. 410901

Волков А. С.

Проверил: Усенко Ф. В.

Минск 2024

Цель: сформировать навыки и умения обработки структурированных типов данных, организованных в виде функций.

Задание:

1. Среди всех четырехзначных номеров машин, определите количество номеров машин, содержащих только две одинаковые цифры.
2. Используя перегрузку методов, создайте программу согласно варианту.

а) для умножения вещественных чисел;

б) для умножения комплексных чисел.

Код программы для задачи 1 :

#include <iostream>

using namespace std;

#include <sstream>

// Функция проверки, содержит ли номер ровно две одинаковые цифры

bool hasExactlyTwoSameDigits(const string& number) {

int digitCount[10] = { 0 }; // Массив для подсчета количества каждой цифры

// Подсчитываем количество каждой цифры

for (char digit : number) {

digitCount[digit - '0']++;

}

// Подсчитываем количество цифр, которые встречаются дважды

int pairs = 0;

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

if (digitCount[i] == 2) {

pairs++;

}

}

// Если ровно две одинаковые цифры, значит есть одна пара и остальные цифры уникальны

return pairs == 1;

}

// Основная функция

int countValidCarNumbers() {

int count = 0;

// Перебираем все четырехзначные номера

for (int i = 1000; i <= 9999; i++) {

string number = to\_string(i); // Преобразуем число в строку

if (hasExactlyTwoSameDigits(number)) {

count++;

}

}

return count;

}

int main() {

setlocale(LC\_ALL, "Ru");

int result = countValidCarNumbers();

cout << "Количество номеров машин с ровно двумя одинаковыми цифрами: " << result << endl;

return 0;

}

Код программы для задачи 2 :

#include <iostream>

#include <cmath>

using namespace std;

// Перегруженная функция для умножения вещественных чисел

double multiply(double a, double b) {

return a \* b;

}

// Перегруженная функция для умножения комплексных чисел

void multiply(double real1, double imag1, double real2, double imag2, double& resultReal, double& resultImag) {

resultReal = real1 \* real2 - imag1 \* imag2; // Реальная часть

resultImag = real1 \* imag2 + imag1 \* real2; // Мнимая часть

}

int main() {

setlocale(LC\_ALL, "RU");

double real1, imag1 = 0.0; // Первое число

double real2, imag2 = 0.0; // Второе число

cout << "Введите первое число (если комплексное, укажите мнимую часть через пробел без i): ";

cin >> real1;

if (cin.peek() != '\n') { // Проверяем, есть ли ввод второй части

cin >> imag1;

}

cout << "Введите второе число (если комплексное, укажите мнимую часть через пробел без i): ";

cin >> real2;

if (cin.peek() != '\n') { // Проверяем, есть ли ввод второй части

cin >> imag2;

}

// Проверяем, являются ли числа вещественными или комплексными

if (imag1 == 0.0 && imag2 == 0.0) {

// Вещественные числа

cout << "Результат умножения вещественных чисел: " << multiply(real1, real2) << endl;

}

else {

// Комплексные числа

double resultReal, resultImag;

multiply(real1, imag1, real2, imag2, resultReal, resultImag);

cout << "Результат умножения комплексных чисел: (" << real1 << " + " << imag1 << "i) \* ("

<< real2 << " + " << imag2 << "i) = " << resultReal << " + " << resultImag << "i" << endl;

}

return 0;

}

На рисунке показаны скриншоты работающей 1 программы.

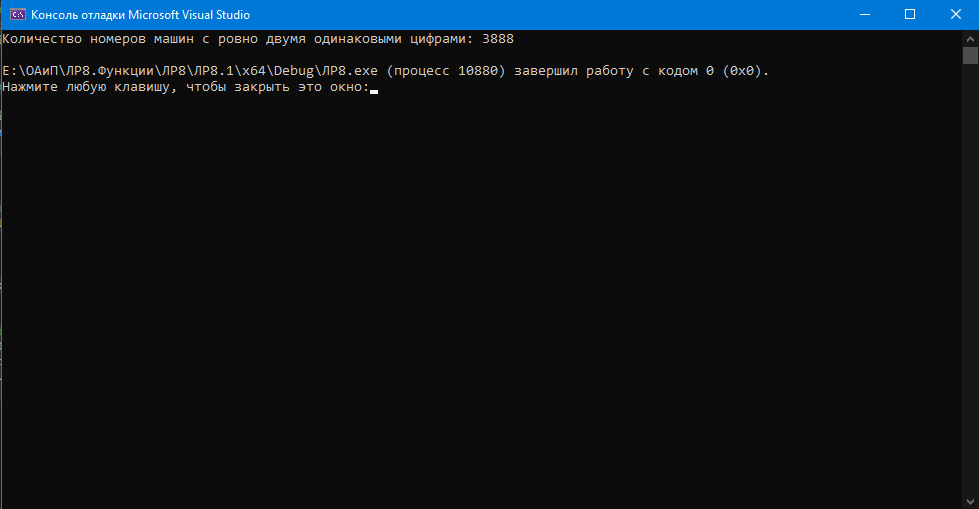


Рисунок 1 – Скриншот консоли с демонстрацией работы функции

На рисунках 1-2 показаны скриншоты работающей 2 программы.

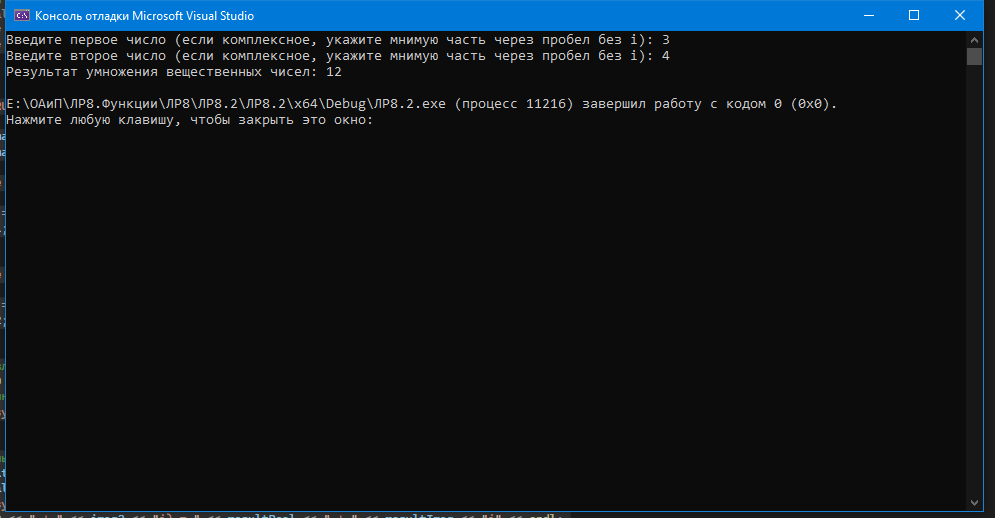
т

Рисунок 1 –Демонстрация работы программы с вещественными числами

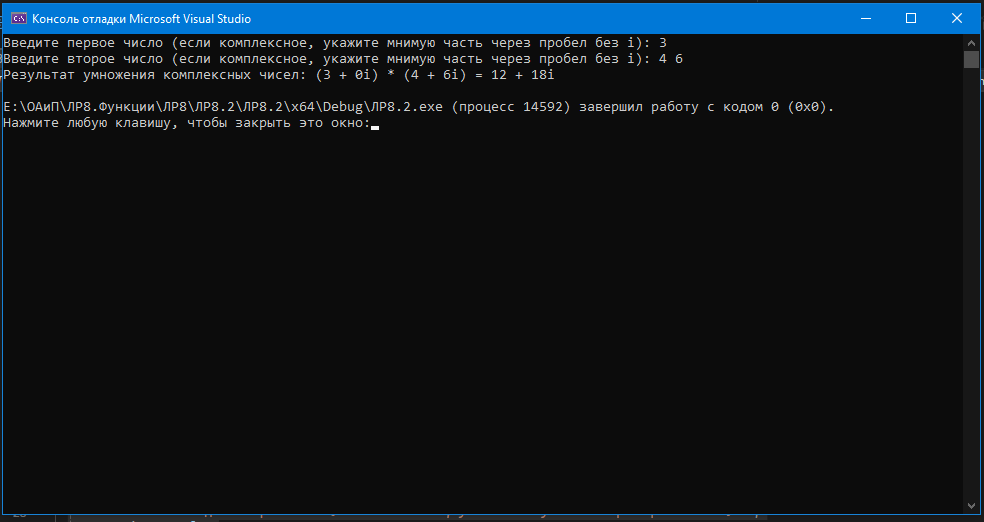
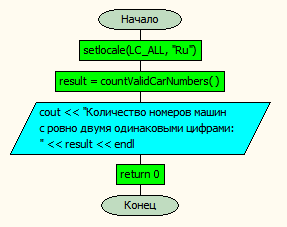


Рисунок 2 – Демонстрация работы программы с комплексными числами

**Вывод:** В ходе выполнения лабораторной работы были сформированы навыки и умения обработки данных, организованных в виде функций. Были изучены методы создания и вызова функций, передача параметров, возвращение результатов, а также организация программы с использованием функций для повышения ее читаемости и модульности. Полученные знания и навыки будут полезны для разработки структурированных и легко сопровождаемых программ.

Построение блок-схем:

1.

2.