\Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Оренбургский государственный университет»

Кафедра программного обеспечения вычислительной техники и
автоматизированных систем

Л.Ф. Тагирова, Е.Н. Чернопрудова

РЕШЕНИЕ ПРАКТИЧЕСКИХ ЗАДАЧ СТРУКТУРНОГО ПРОГРАММИРОВАНИЯ. ЧАСТЬ 2

Методические указания

Рекомендовано к изданию редакционно-издательским советом федерального государственного бюджетного образовательного учреждения высшего образования «Оренбургский государственный университет» для обучающихся по образовательным программам высшего образования по направлениям подготовки 09.03.01 Информатика и вычислительная техника и 09.03.04 Программная инженерия

Оренбург
2019

УДК 004.422.63(076.5)
ББК 32.973-018я7
Т 13

Рецензент – доктор технических наук, профессор Т.М. Зубкова

Тагирова, Л. Ф.
Т 13 Решение практических задач структурного программирования. Часть 2: методические указания / Л.Ф. Тагирова, Е.Н. Чернопрудова; Оренбургский гос. унт. Оренбург: ОГУ, 2019. – 80 с.

Методические указания содержат рекомендации для проведения практических занятий по разработке программ на языке программирования высокого уровня С++. Описывается создание собственных функций пользователя, приложений с использованием графических интерфейсов. Помимо этого в методических указаниях представлены задания по работе со структурами и объединениями, динамическими структурами данных, файлами, а также использование элементов объектно-ориентированного программирования при разработке приложений.

Методические указания предназначены для обучающихся по направлениям подготовки: 09.03.01 Информатика и вычислительная техника при изучении дисциплины «Программирование» и 09.03.04 Программная инженерия при изучении дисциплины «Программирование и алгоритмизация».

УДК 004.422.63(076.5)
ББК 32.973-018я7

Содержание

Введение

Язык C++ в настоящее время является одним из наиболее распространенных языков программирования в мире. C++ является компилируемым, статически типизированным языком программирования общего назначения. Язык C++ поддерживает такие парадигмы программирования как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Помимо этого C++ обеспечивает модульность, использует обработку исключений, объявление типов (классов) объектов, виртуальные функции [1].

Стандартная библиотека языка C++ включает, в том числе, общеупотребительные контейнеры и алгоритмы. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков. В сравнении с его предшественником языком C, в языке C++ наибольшее внимание уделено поддержке объектно-ориентированного и обобщённого программирования [2].

C++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также развлекательных приложений [3].

Целью данных методических указаний является обучение студентов разработке сложных программ на языке C++ с использованием расширенных инструментов языка.

Представленные методические указания состоит из введения, семи глав, включающих описание практических работ, заключения, списка использованных источников, приложения.

В первой главе рассматривается работа со структурами и объединениями в языке C++.

Во второй главе представлено задание на работу с динамическими структурами данных.

В третьей главе представлена практическая работа по созданию собственных функций пользователя с помощью языка C++.

В следующей главе, номер четыре, рассматривается практическая работа по работе с файлами.

Далее, в пятой главе, рассматривается создание приложений с использованием графических интерфейсов приложений с помощью языка C++.

В следующей главе, номер шесть, рассматривается использование элементов объектно-ориентированного программирования при разработке приложений на языке C++.

В седьмой главе представлено задание на практическую работу по созданию приложений с использованием стандартных библиотек на языке C++.

В приложении представлена структура оформления отчета к практическим заданиям.

После окончания изучения представленных методических указаний студенты будут уметь самостоятельно разрабатывать приложения на языке высоко уровня C++.

Авторы выражают глубокую благодарность и признательность рецензенту за внимательное прочтение рукописи и замечания, способствовавшие улучшению качества предлагаемых методических указаний.

1 Практическое занятие №1. Работа со структурами и объединениями

Цель: Закрепление теоретических знаний и получение практических навыков создания данных структурного (пользовательского) типа, освоить приёмы обработки массивов структур.

Задание. Составить программу с использованием одного из типов данных, определяемых пользователем: структуры (struct).

Вариант 1. Необходимо описать структуру с именем APPLIANCES, содержащую следующие поля:

- FIRM – название фирмы;
- APPLICATION - применение.
- COUNTRY – страна-производитель.

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из восьми элементов типа APPLIANCES; записи должны быть отсортированы по названию фирмы в алфавитном порядке;
- выводить на экран информацию о бытовой технике, относящихся к одному и тому же виду применения. Вид применения вводить с клавиатуры;
- если такой бытовой техники нет, выводить соответствующее сообщение на экран.

Вариант 2. Необходимо описать структуру с именем PHILOSOPHER, содержащую следующие поля:

- NAME - фамилия, имя;
- DAY - день рождения (массив из трех чисел).
- PERIOD - период существования.

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из восьми элементов типа PHILOSOPHER; записи должны быть отсортированы по фамилии в алфавитном порядке;

- выводить на экран информацию о философах, существовавших в один период времени. Период времени вводить с клавиатуры;

- если таких писателей нет, выводить соответствующее сообщение на экран.

Вариант 3. Необходимо описать структуру с именем WRITER, содержащую следующие поля:

- NAME - фамилия, имя;

- DAY - день рождения (массив из трех чисел).

- STYLE - стиль.

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из восьми элементов типа WRITER; записи должны быть отсортированы по фамилии в алфавитном порядке;

- выводить на экран информацию о писателях, пишущих в одном стиле. Стиль вводить с клавиатуры;

- если таких писателей нет, выводить соответствующее сообщение на экран.

Вариант 4. Необходимо описать структуру с именем TRANSLATOR, содержащую следующие поля:

- NAME - фамилия, имя;

- DAY - день рождения (массив из трех чисел).

- LANGUAGE – язык.

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из восьми элементов типа TRANSLATOR; записи должны быть отсортированы по фамилии в алфавитном порядке;

- выводить на экран информацию о переводчиках, работающих с одним языком при переводе. Язык вводить с клавиатуры;

- если таких переводчиков нет, выводить соответствующее сообщение на экран.

Вариант 5. Необходимо описать структуру с именем PROGRAMMER, содержащую следующие поля:

- NAME - фамилия, имя;

- DAY - день рождения (массив из трех чисел).

- LANGUAGE – язык программирования.

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из восьми элементов типа PROGRAMMER; записи должны быть отсортированы по фамилии в алфавитном порядке;

- выводить на экран информацию о программистах, программирующих на одном языке программирования. Язык программирования вводить с клавиатуры;

- если таких программистов нет, выводить соответствующее сообщение на экран.

Вариант 6. Необходимо описать структуру с именем PILOT, содержащую следующие поля:

- NAME - фамилия, имя;

- DAY - день рождения (массив из трех чисел).

- YEAR - год полета.

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из восьми структур типа PILOT; записи должны быть отсортированы по фамилии в алфавитном порядке;

- выводить на экран информацию о летчиках, отправившихся в полет в год введенный с клавиатуры;

- если таких летчиков нет, выводить соответствующее сообщение на экран.

Вариант 7. Необходимо описать структуру с именем COSMONAUT, содержащую следующие поля:

- NAME - фамилия, имя;

- DAY - день рождения (массив из трех чисел).

- YEAR - год полета.

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из восьми элементов типа COSMONAUT; записи должны быть отсортированы по фамилии в алфавитном порядке;

- выводить на экран информацию о космонавтах, отправившихся в космос в год введенный с клавиатуры;

- если таких космонавтов нет, выводить соответствующее сообщение на экран.

Вариант 8. Необходимо описать с именем SPORTSMAN, содержащую следующие поля:

- NAME - фамилия, имя;

- DAY - день рождения (массив из трех чисел).

- SPORT – вид спорта;

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из шести элементов типа SPORTSMAN; записи должны быть отсортированы по фамилии в алфавитном порядке;

- выводить на экран информацию о спортсменах, занимающиеся одним видом спорта. Вид спорта необходимо вводить с клавиатуры;

- если таких спортсменов нет, выводить соответствующее сообщение на экран.

Вариант 9. Необходимо описать структуру с именем SCIENTIST, содержащую следующие поля:

- NAME - фамилия, имя;

- DAY - день рождения (массив из трех чисел).

- STUDY – направление в науки;

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из восьми элементов типа SCIENTIST; записи должны быть упорядочены по изучаемому направлению науки;

- выводить на экран информацию об ученых, фамилия которых начинается на букву «А»;

- если таких ученых нет, выводить соответствующее сообщение на экран.

Вариант 10. Необходимо описать структуру с именем MUSICIAN, содержащую следующие поля:

- NAME - фамилия, имя;

- PERIOD - период;

- STYLE - стиль.

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из восьми элементов типа MUSICIAN; записи должны быть упорядочены по стилю музыки;

- выводить на экран информацию о музыкантах, одного периода времени, Период необходимо вводить с клавиатуры;

- если таких музыкантов нет, выводить соответствующее сообщение на экран.

Вариант 11. Необходимо описать структуру с именем PAINTER, содержащую следующие поля:

- NAME - фамилия, имя;

- PERIOD - период;

- STYLE - стиль.

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив SPISOK, состоящий из восьми элементов типа PAINTER; записи должны быть упорядочены по стилю рисованию;

- выводить на экран информацию о художниках, фамилия которых начинается на букву «М»;

- если таких художников нет, выводить соответствующее сообщение на экран.

Вариант 12. Необходимо описать структуру с именем ACTOR, содержащую следующие поля:

- NAME - фамилия;

- LIVE – страна рождения;

- DAY - день рождения (массив из трех чисел).

Разработать приложение, выполняющее следующие действия:

- вводить с клавиатуры, данные в массив SPISOK, состоящий из восьми элементов типа ACTOR; записи должны быть упорядочены по датам дней рождения;

- выводить на экран информацию об актерах, рожденных в один год. Год рождения вводить с клавиатуры;

- если таких актеров нет, выводить соответствующее сообщение на экран.

Вариант 13. Необходимо описать структуру с именем GALAXY, содержащую следующие поля:

- NAME - название;

- TYPE – тип галактики;

- CORE – яркость ядра.

Разработать приложение, выполняющее следующие действия:

- вводить с клавиатуры, данные в массив SPISOK, состоящий из восьми элементов типа GALAXY; записи должны быть размещены в алфавитном порядке по названию;

- выводить на экран информацию о галактиках, относящихся к одному типу. Тип галактики вводить с клавиатуры;

- если таких галактик нет, выводить соответствующее сообщение на экран.

Вариант 14. Необходимо описать структуру с именем PEOPLE, содержащую следующие поля:

- NAME - фамилия;

- PASPORT - номер паспорта;

- DAY - день рождения (массив из трех чисел).

Разработать приложение, выполняющее следующие действия:

- вводить с клавиатуры, данные в массив SPISOK, состоящий из восьми элементов типа PEOPLE; записи должны быть размещены в алфавитном порядке по фамилии;

- выводить на экран информацию о людях, рожденных в один год. Год рождения вводить с клавиатуры;

- если таких языков нет, выводить соответствующее сообщение на экран.

Вариант 15. Необходимо описать структуру с именем LANGUAGE, содержащую следующие поля:

- NAME – название языка;

- COUNTRY – страна происхождения;

- YEAR – год возникновения.

Разработать приложение, выполняющее следующие действия:

- вводить с клавиатуры, данные в массив SPISOK, состоящий из восьми элементов типа LANGUAGE; записи должны быть размещены в алфавитном порядке по названию языка;

- выводить на экран информацию о языках, расположенных в заданной стране. Название страны вводить с клавиатуры;

- если таких языков нет, выводить соответствующее сообщение на экран.

Вариант 16. Необходимо описать структуру с именем STAR, содержащую следующие поля:

- NAME – название звезды;

- CONSTELLATION – названия созвездия, в котором расположена звезда;

- SIZE – видимая величина.

Разработать приложение, выполняющее следующие действия:

- вводить с клавиатуры, данные в массив SPISOK, состоящий из восьми элементов типа STAR; записи должны быть размещены в алфавитном порядке по названию звезд;

- выводить на экран информацию о звездах, расположенных в созвездие Большая медведица, введенное с клавиатуры;

- если таких звезд нет, выводить соответствующее сообщение на экран.

Вариант 17. Необходимо описать структуру с именем ZODIAC, содержащую следующие поля:

- NAME - фамилия, имя;

- ZNAK- знак Зодиака;

- DAY - день рождения (массив из трех чисел).

Разработать приложение, выполняющее следующие действия:

- вводить с клавиатуры, данные в массив SPISOK, состоящий из восьми элементов типа ZODIAC; записи должны быть упорядочены по знакам Зодиака;

- вывод на экран информации о людях, родившихся в месяц, значение которого введено с клавиатуры;

- если таких нет, выводить соответствующее сообщение на экран.

Вариант 18. Необходимо описать структуру с именем CAT, содержащую следующие поля:

- BREED - название породы;

- COLOR – цвет;

- STOIM - стоимость породы в руб.

Разработать приложение, выполняющее следующие действия:

- вводить с клавиатуры, данные в массив SPISOK, состоящий из восьми элементов типа CAT. Записи должны быть размещены в алфавитном порядке

по названиям пород;

- выводить на экран информации о кошках, название породы которой введено с клавиатуры;

- если таких пород нет, выводить соответствующее сообщение на экран.

Вариант 19. Необходимо описать структуру с именем DOG, содержащую следующие поля:

- BREED - название породы;

- COLOR – цвет;

- STOIM - стоимость породы в руб.

Разработать приложение, выполняющее следующие действия:

- вводить с клавиатуры, данные в массив SPISOK, состоящий из восьми элементов типа DOG. Записи должны быть размещены в алфавитном порядке

по названиям пород;

- выводить на экран информации о собаках, название породы которой введено с клавиатуры;

- если таких пород нет, выводить соответствующее сообщение на экран.

Вариант 20. Необходимо описать структуру с именем PLANET, содержащую следующие поля:

- NAME – название планеты;

- COLOR – цвет планеты;

- LENGHT – длина экватора.

Разработать приложение, выполняющее следующие действия:

- вводить с клавиатуры, данные в массив SPISOK, состоящий из восьми элементов типа PLANET; записи должны быть размещены в алфавитном порядке по названию планет;

- выводить на экран информацию о планете, голубого цвета, введенного с клавиатуры;

- если таких планет нет, выводить соответствующее сообщение на экран.

Пример выполнения практического задания №1.

Задание.

Необходимо описать структуру с именем TRAIN, содержащую следующие поля:

- NAZN - название пункта назначения;

- NUMR - номер поезда;

- TIME - время отправления.

Разработать приложение, выполняющее следующие действия:

- ввод с клавиатуры данных в массив RASP, состоящий из восьми элементов типа TRAIN;

- вывод на экран информации о поездах, отправляющихся в веденное с клавиатуры временя;

- если таких поездов нет, выводить соответствующее сообщение на экран.

Решение.

1) Словесный алгоритм.

Задана структура TRAIN. В данной структуре имеется три поля: NAZN – типа string, NUMR – типа int и TIME типа double. Ввод данных в структуру осуществлялся с помощью цикла for. Для поиска поездов которые отправляются в веденное время каждый элемент массива RASP проверялся по полю TIME. В случае совпадения на экран выводиться информация о данном поезде, иначе выводиться сообщение об ошибке.

2) Схема алгоритма. Схема алгоритма программы представлена на рисунке 1.

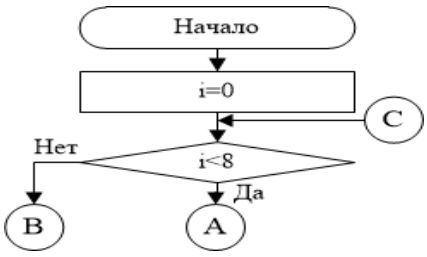

Рисунок 1, Лист 1 - Схема алгоритма реализации работы со структурами данных

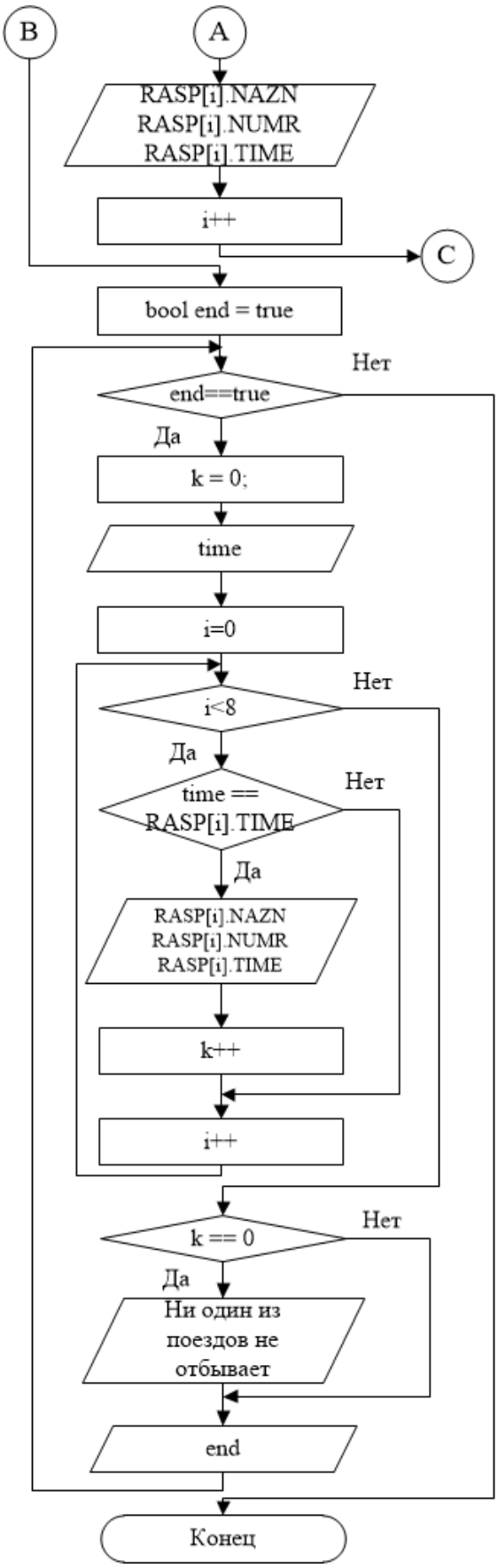

Рисунок 1, Лист 2

3) Код программы.

```
#include "pch.h"
#include <iostream>
#include <string>
using namespace std;
struct TRAIN
{
string NAZN;      //Пункт назначения
int NUMR;         //Номер поезда
double TIME;      //Время отправления
};
int main() {
setlocale(LC_ALL, "Rus");   //Установка русского языка в консоли
int i,k=0;
double time;
TRAIN RASP[8];                  // Объявление массива RASP типа TRAIN
cout << "-----------------------------------" << "\nВбейте нужные значения о поездах в базу данных:";
for ( i = 0; i < 8; i++) {
cout << "\nДля " << i + 1 << " поезда из 8: ";
cout << "\nПункт назначения: ";
cin >> RASP[i].NAZN;                        /*
cout << "Номер поезда: ";             Ввод информации о поездах
cin >> RASP[i].NUMR;
cout << "Время отправления: ";
cin >> RASP[i].TIME;                        */
if (i<7)   // Разделение данных в консоли для наглядности
cout << ")";
 }
```

```
cout << "----------------------------------------";
bool end = true;
while (end)
{
k = 0;
cout << "\nВремя отправления: ";
cin >> time;
for (i = 0; i < 8; i++) {
if (time == RASP[i].TIME)    // Поиск совпадений по времени
{
cout << "Пункт назначения " << RASP[i].NAZN << " Поезд № " << RASP[i].NUMR << " Время отбытия [" << RASP[i].TIME << "]";    // Вывод информации о поезде
k++;
}
}
if (k == 0)
cout << "\nНи один из поездов не отбывает в " << time << endl; //Сообщение об ошибке
cout << "\nВвести время еще раз? (1-да|0-нет)" << endl; // Повторный поиск
cin >> end;
}
cout << endl<<endl;
}
```

4) Реализация программного кода. Реализация программного кода представлена на рисунках 2 и 3.

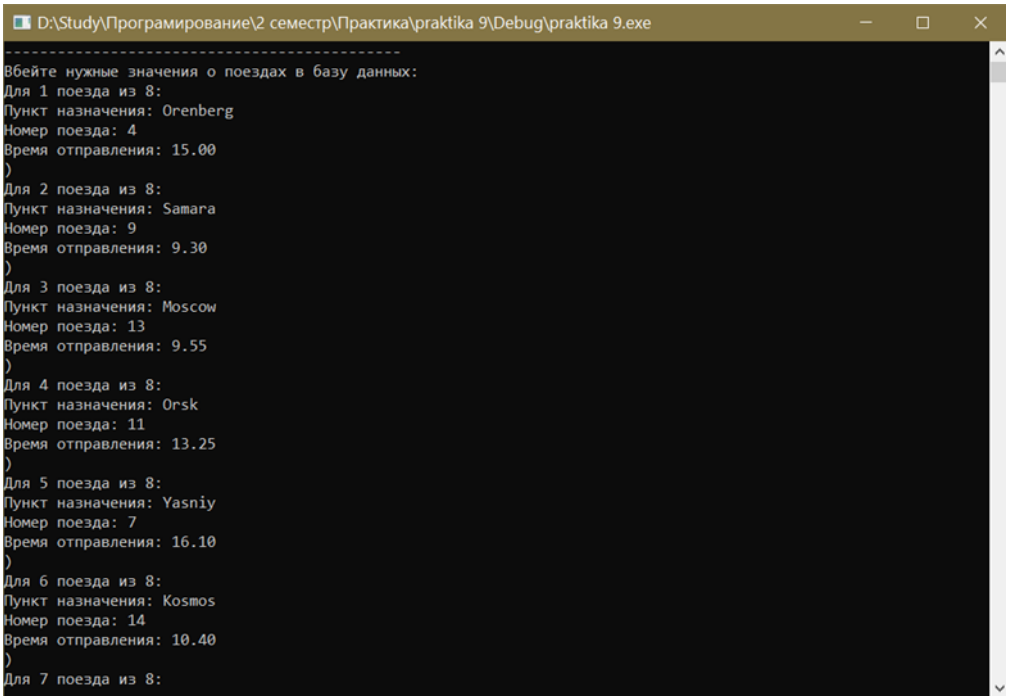

Рисунок 2 – Результат работы программы со структурами и объединениями

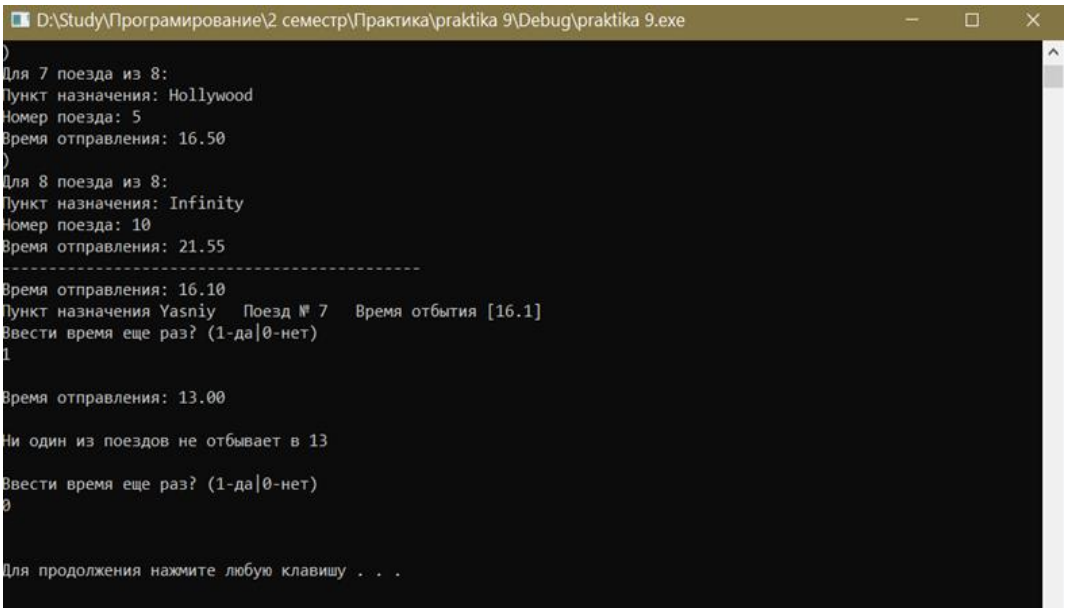

Рисунок 3 – Продолжение результата работы программы со структурами и объединениями

2 Практическое занятие №2. Динамические структуры данных

Цель: Закрепление теоретических знаний и получение практических навыков создания приложений с использованием динамических структур данных.

Задание: Составить программу с использованием одного из способов реализации динамической структуры данных (линейные списки, стеки, очереди или бинарные деревья).

Вариант 1. Необходимо составить программу, которая будет обрабатывать входной файл, и составлять для него алфавитно-частотный словарь в выходном файле. Результат работы программы вывести на экран.

Вариант 2. Необходимо составить программу, в которую требуется ввести символьную строку. В строке должны содержаться три вида скобок:

1) круглые;
2) квадратные;
3) фигурные.

Требуется определить, верно ли расставлены скобки. Не учитывать символы между строками. Результат работы программы вывести на экран.

Вариант 3. Необходимо составить программу, в которую требуется ввести два целых числа m и n. Найти все делители числа n, взаимно простые с числом m. Результат работы программы вывести на экран.

Вариант 4. Необходимо составить программу, в которую требуется ввести последовательность ненулевых целых чисел. В которой число 0 - признак конца последовательности. Требуется найти первый наибольший отрицательный элемент. В случаи отсутствия такого элемента необходимо вывести на экран сообщение о его отсутствие.

Вариант 5. Необходимо составить программу, в которую требуется ввести последовательность целых чисел. Требуется определить, количество чисел,

которые встречаются в последовательности только один раз. Вывести все найденные числа на экран.

Вариант 6. Необходимо составить программу, в которую требуется ввести число, затем перевести введенное с клавиатуры число в систему счисления с заданным основанием. Результат работы программы вывести на экран.

Вариант 7. Необходимо составить программу, в которую требуется поочередно ввести элементы очереди состоящей из 9 элементов. Вывести ее на экран, а затем поочередно удалить все элементы очереди, после удаления всех элементов вывести сообщение о том, что очередь пуста.

Вариант 8. Необходимо составить программу, в которую требуется ввести символьную строку. В строке должны содержаться квадратные скобки. Требуется определить, верно ли расставлены скобки. Результат работы программы вывести на экран.

Вариант 9. Необходимо составить программу, которая будет обрабатывать входной файл, состоящий из списка чисел. Требуется переписать список в выходной файл в обратном порядке. Результат работы программы вывести на экран.

Вариант 10. Необходимо составить программу, которая будет производить вычисления правильного арифметического выражения содержащего целые числа и скобки. Результат работы программы вывести на экран.

Вариант 11. Необходимо составить программу, в которой дан указатель P_1 на вершину стека, содержащего не менее 10 элементов. Затем извлечь из стека первые девять элементов и вывести их значения на экран. А также вывести адрес новой вершины стека. После извлечения элементов из стека освобождать память, которую они занимали.

Вариант 12. Необходимо составить программу, в которую требуется ввести список. Затем упорядочить список по убыванию. Далее необходимо вставить новое значение, которое должно вводиться с клавиатуры. Новое значение не

должно нарушать упорядоченность списка. Результат работы программы вывести на экран.

Вариант 13. Необходимо составить программу, в которой даны указатели P_1 и P_2 на вершины двух непустых стеков. Требуется переместить все элементы из первого стека во второй, а затем необходимо вывести адрес новой вершины второго стека на экран.

Вариант 14. Необходимо составить программу, в которую требуется ввести символьную строку. В строке должны содержаться круглые скобки. Требуется определить, верно ли расставлены скобки. Результат работы программы вывести на экран.

Вариант 15. Необходимо составить программу, в которой требуется ввести последовательность слов и вывести на экран. Далее необходимо подсчитать частоту встречаемости каждого слова. Результат работы программы вывести на экран.

Вариант 16. Необходимо составить программу, которая будет обрабатывать входной файл, состоящий из списка слов. Требуется переписать список в выходной файл в обратном порядке. Результат работы программы вывести на экран.

Вариант 17. Необходимо составить программу, в которую требуется ввести символьную строку. В строке должны содержаться фигурные скобки. Требуется определить, верно ли расставлены скобки. Результат работы программы вывести на экран.

Вариант 18. Необходимо составить программу, в которую требуется ввести список. Затем упорядочить список по возрастанию. Далее необходимо вставить новое значение, которое должно вводиться с клавиатуры. Новое значение не должно нарушать упорядоченность списка. Результат работы программы вывести на экран.

Вариант 19. Необходимо составить программу, которая будет обрабатывать входной файл, состоящий из списка чисел. Требуется переписать список в выходной файл в порядке возрастания. Результат вывести на экран.

Вариант 20. Необходимо составить программу, в которой дан набор из 10 чисел. Требуется создать очередь, содержащую данные числа. Первое число очереди должно размещаться в начале, а последнее число в конце очереди. Необходимо вывести указатели P_1 и P_2 на начало и конец очереди. Результат работы программы вывести на экран.

Пример выполнения практического задания №2.

Задание.

Необходимо составить программу, в которую требуется ввести символьную строку. В строке должны содержаться фигурные скобки. Требуется определить, верно ли расставлены скобки. Результат работы программы вывести на экран.

Решение.

1) Словесный алгоритм.

Была объявлена глобальная структура "comp", содержащая символьный элемент и указатель на следующий элемент стека. Далее заданы две void функции. Первая считывает элемент и создает новый элемент в стеке, вторая принимала созданный ранее стек и 3 переменных типа int (в ней вычислялось количество скобок различных типов). Далее в main функции считывается строка, переменные, содержащие количество скобок, обнулялись, и в цикле for все символы кроме скобок пропускались, а скобки добавлялись в стек. Далее вызывается функция, которая считывает количество скобок различного типа, а так же вид спаривает скобки, и в случае несовпадения на экран выводилось сообщение: «Закрывающая скобка другого типа». Если же не совпадало количество закрывающих и открывающих скобок, то на экран выводиться сообщение: «Лишняя закрывающая скобка». В конце работы функции выводиться надпись, верно ли были расставлены скобки.

2) Схема алгоритма. Схема алгоритма программы представлена на рисунках 4-6.

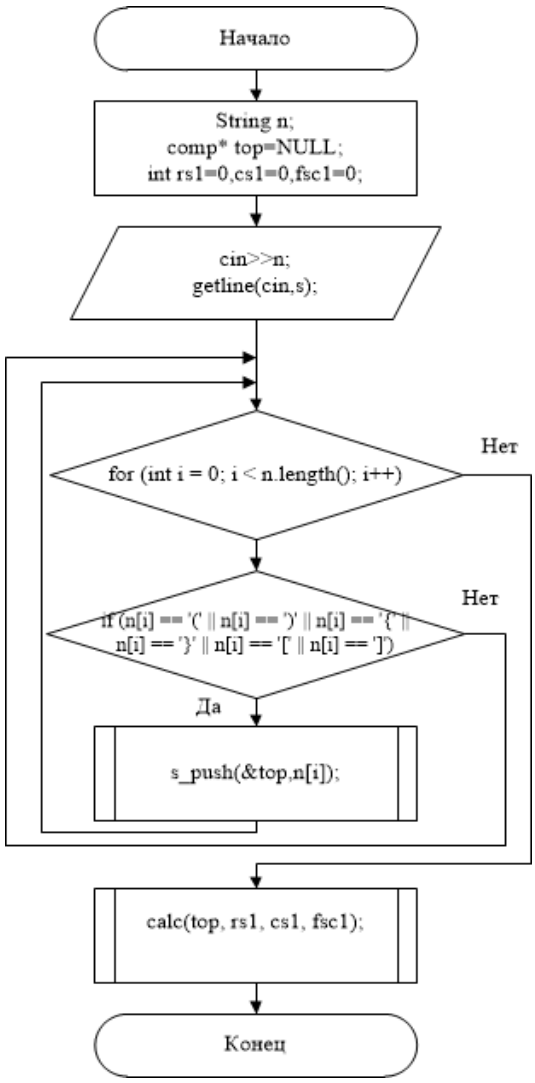

Рисунок 4 - Схема алгоритма реализации работы с динамическими структурами данных

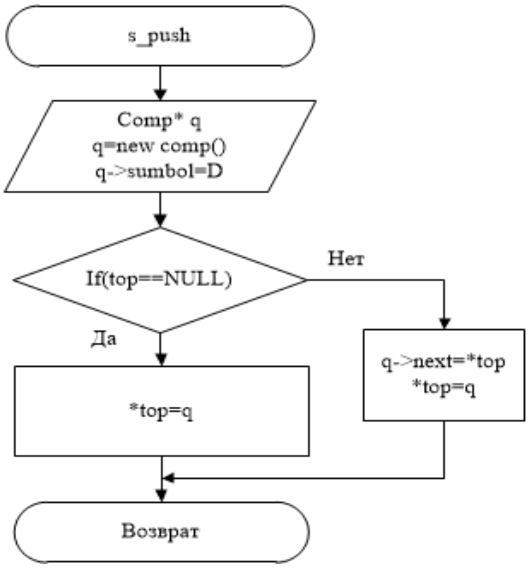

Рисунок 5 – Схема алгоритма функции s_push

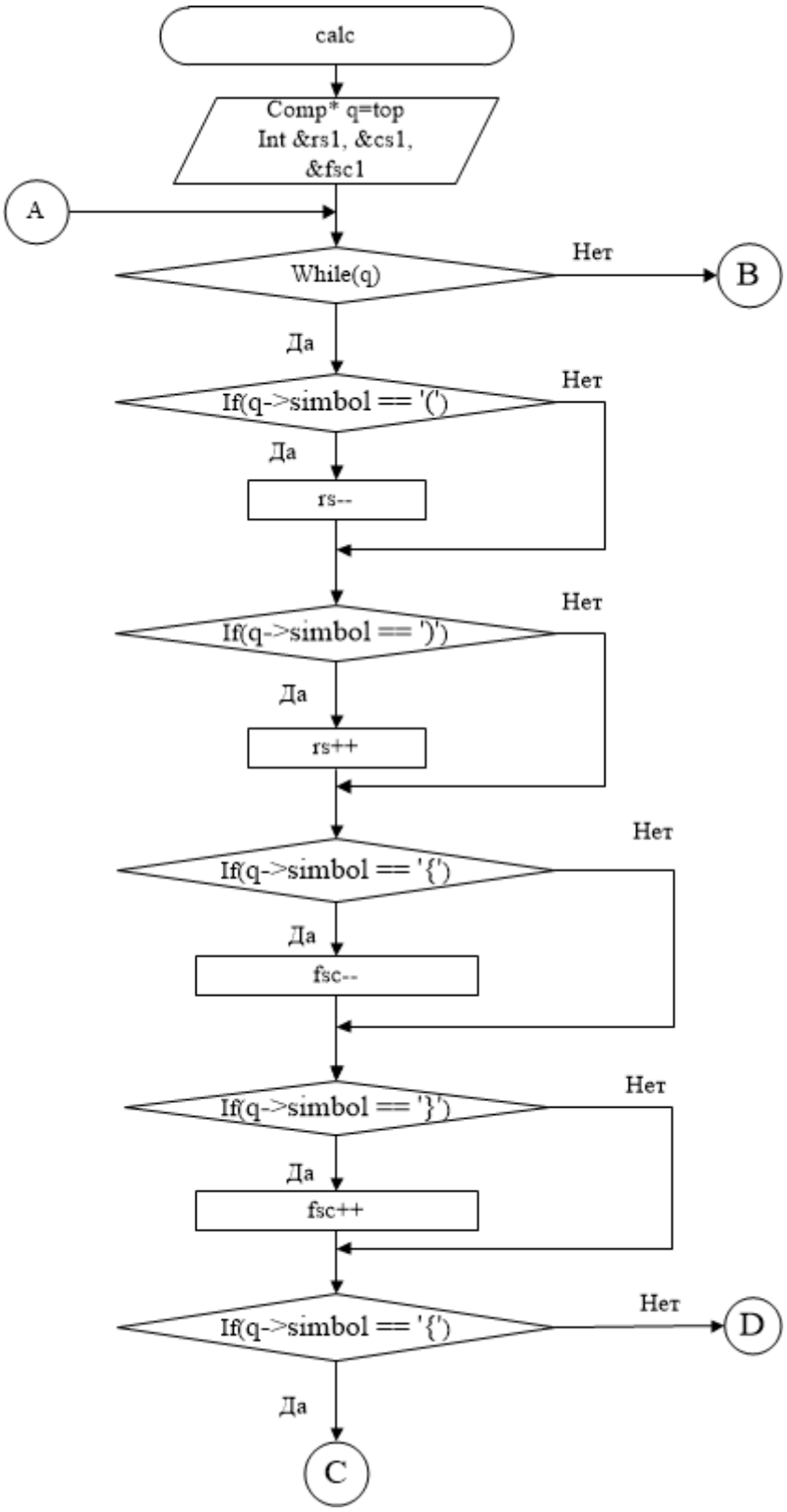

Рисунок 6, Лист 1 – Схема алгоритма функции calc

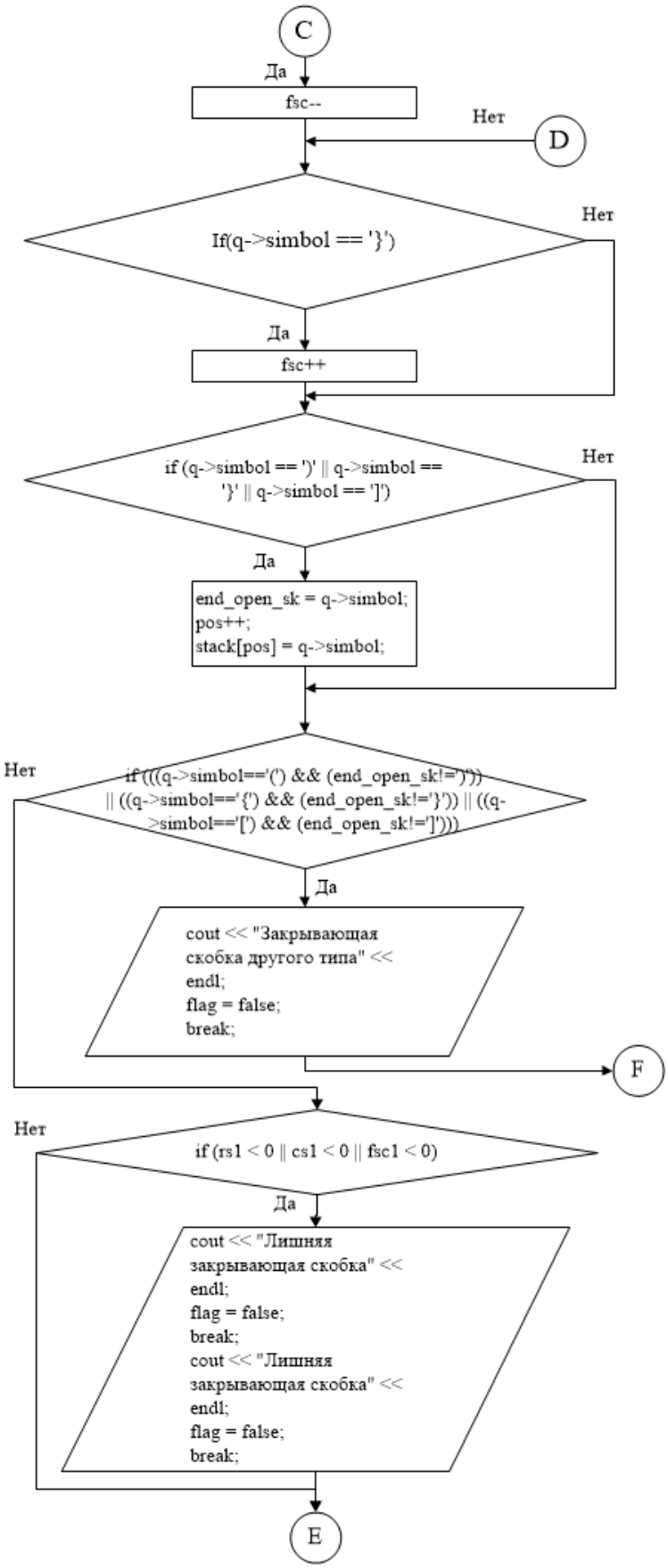

Рисунок 6, Лист 2

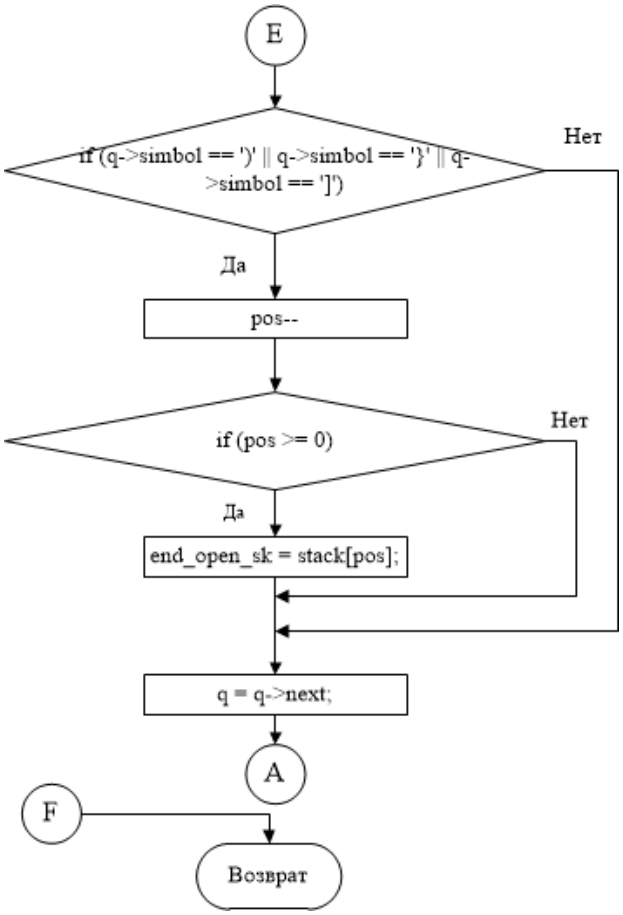

Рисунок 6, Лист 25

3) Код программы.

```
#include "pch.h"
#include <iostream>
#include <string>
#include <vector>
using namespace std;
struct comp
{
char simbol;
comp* next;
};
vector<char> stack;
```

```
char end_open_sk;
int pos = -1;
bool flag = true;
void s_push(comp** top,char D) {
comp* q;
q = new comp();
q->simbol = D;
if (top == NULL) {
*top = q; }
else {
q->next = *top;
*top = q; }
}
void calc(comp* top,int& rs1,int& cs1,int& fsc1) {
comp *q = top;
while (q) {
if (q->simbol == '(') rs1--;              // Счетчик круглых скобок
if (q->simbol == ')') rs1++;
if (q->simbol == '{') fsc1--;             // Счетчик фигурных скобок
if (q->simbol == '}') fsc1++;
if (q->simbol == '[') cs1--;               // Счетчик квадратных скобок
if (q->simbol == ']') cs1++;
if (q->simbol == ')' || q->simbol == '}' || q->simbol == ']') {
```

end_open_sk = q->simbol; //Т. К. стек развернут относительно оригинальной строки , то закрывающие скобки будут открывающими, в данном if в массив записываются открывающие скобки для последующей проверки

```
pos++;
stack[pos] = q->simbol; }
```

```
if (((q->simbol=='(') && (end_open_sk!=')')) || ((q->simbol=='{') && (end_open_sk!='}')) || ((q->simbol=='[') && (end_open_sk!=']'))) { //В данном if проверяется соответствие открывающих и закрывающих скобок
cout << "Закрывающая скобка другого типа" << endl;
flag = false;
break; }
if (rs1 < 0 || cs1 < 0 || fsc1 < 0) { //Благодаря значениям подсчитанным ранее сверяем количество скобок всех типов
cout << "Лишняя закрывающая скобка" << endl;
flag = false;
break; }
// Данный if работает из глубины наружу, меняя закрывающую скобку по мере прохождения по стеку
if (q->simbol == ')' || q->simbol == '}' || q->simbol == ']') {
pos--;
if (pos >= 0) {
end_open_sk = stack[pos]; }
}
q = q->next; }
if (!flag) {
cout << "Скобки расставлены неверно" << endl; }
else cout << "Скобки расставлены верно" << endl; }
int main()
{ setlocale(LC_ALL, "Russian");          // Установка русского языка в консоли
string n;
comp *top = NULL;
int rs1, cs1, fsc1;
rs1 = cs1 = fsc1 = 0;
```

```
cout << "Введите строку - ";
getline(cin, n);      //Ввод строки
stack.resize(n.length()); // Объявление размера стека
for (int i = 0; i < n.length(); i++) {
if (n[i] == '(' || n[i] == ')' || n[i] == '{' || n[i] == '}' || n[i] == '[' || n[i] == ']') {
s_push(&top,n[i]); //Вызов функции s_push }
}
calc(top, rs1, cs1, fsc1);                    // Вызов функции calc
return 0; }
```

4) Реализация программного кода. Реализация программного кода представлена на рисунках 7-9.

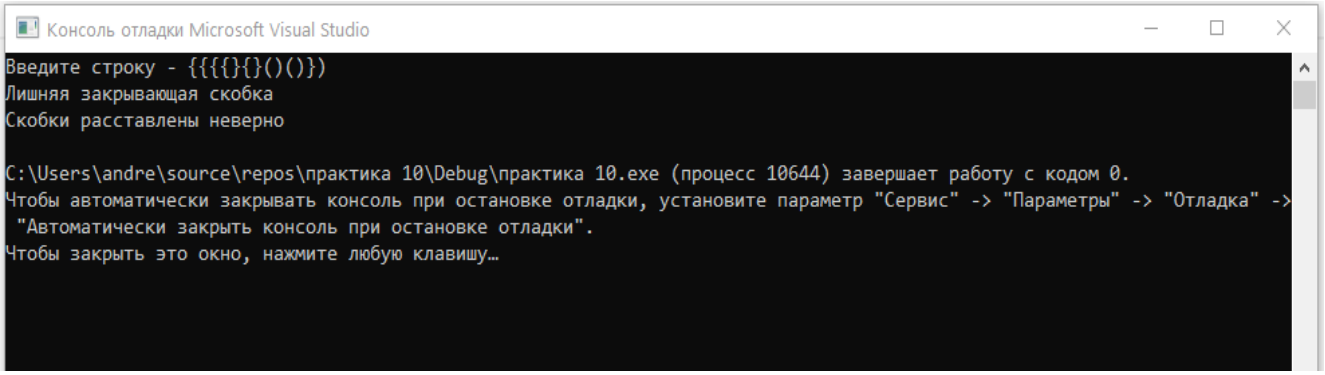

Рисунок 7 – Результат работы программы с динамическими структурами данных

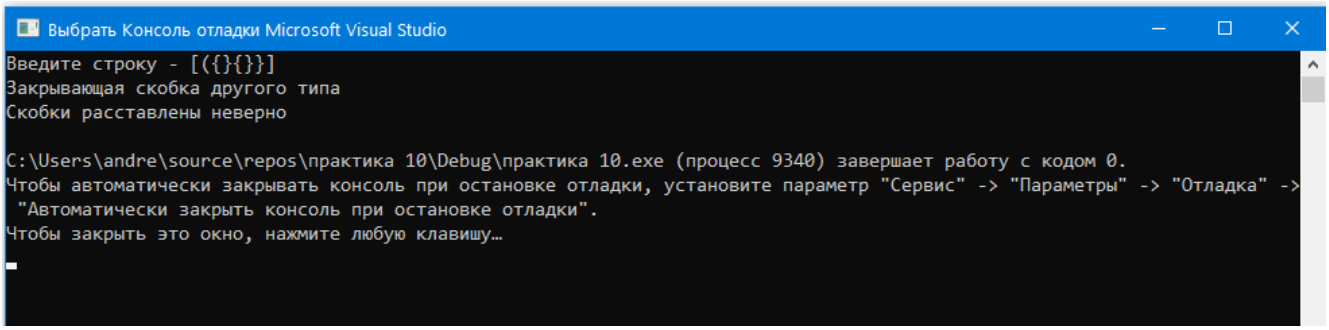

Рисунок 8 – Продолжение результата работы программы. Неверно расставленные скобки

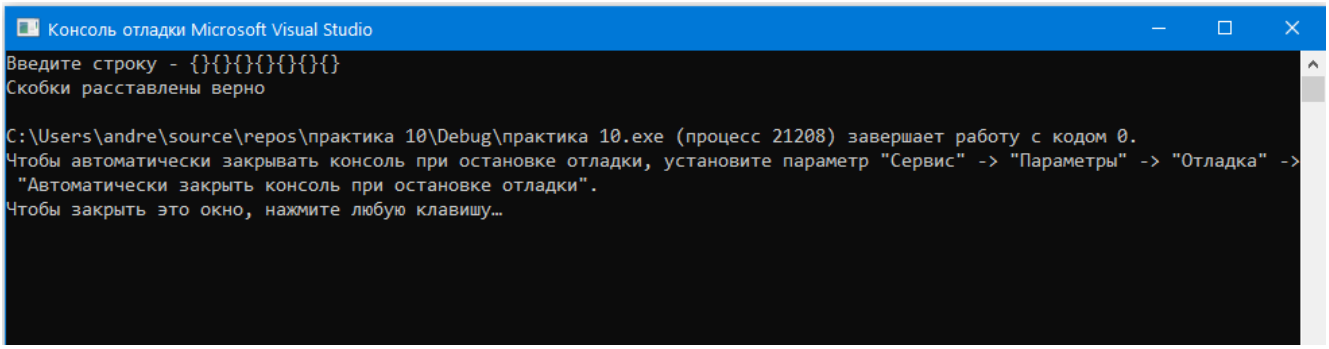

Рисунок 9 – Продолжение результата работы программы. Скобки расставлены верно

3 Практическое занятие №3. Создание собственных функций пользователя в языке C++

Цель: Закрепление теоретических знаний и получение практических навыков создания собственных функций на языке C++.

Задание: Составить программу с использованием пользовательских функций в языке C++.

Вариант 1. Определить функцию, изменяющую исходный вещественный массив путём деления его положительных элементов на свои индексы. Используя эту функцию, определить, в каком из двух массивов: a[22] или b[20] – будет больше изменённых элементов

Вариант 2. Определить одну функцию, которая считает количество элементов до первого положительного в вещественных массивах x[20], y[25], z[20] и формирует из этих элементов новый массив NEW. Вывести на печать исходные и полученные массивы.

Вариант 3. Используя функцию, определить и запомнить в одномерном массиве X номера элементов, значения которых кратны 5 из целочисленных массивов A[20] и B[30]. Вывести на печать исходные и полученный массив.

Вариант 4. Определить одну функцию, вычисляющую номер максимального элемента в одномерном вещественном массиве и количество элементов, стоящих после него. Использовать функцию для массивов W[22] и V[25].

Вариант 5. Определить функцию, которая меняет местами максимальный и минимальный элементы в одномерном вещественном массиве. Использовать функцию для массивов C[15] и D[20]. Вывести на печать исходные и полученные массивы.

Вариант 6. Определить функцию, которая в одномерном массиве заменяет все положительные элементы на 1, а вместо остальных записывает нули. Использовать функцию для массивов P[10] и R[15]. Вывести на печать исходные и полученные массивы.

Вариант 7. Используя одну функцию определить и запомнить в одномерном массиве XX элементы, кратные 3 в массиве X1[20] и элементы, кратные 5 из массива X2[20] подряд. Вывести на печать исходные и полученные массивы.

Вариант 8. Используя одну функцию, сформировать одномерный массив S, состоящий из положительных элементов массива POLOJ[15] и отрицательных элементов массива OTRIC[15]. Вывести на печать исходные и полученные массивы.

Вариант 9. Используя одну функцию, вычислить сумму и количество положительных элементов в вещественных массивах K[9] и E[7]. Вывести на печать исходные и полученные массивы.

Вариант 10. Используя одну функцию, сформировать одномерный массив N, состоящий из чётных элементов целочисленного массива T[15] и нечётных элементов целочисленного массива L[20]. Вывести на печать исходные и полученные массивы.

Вариант 11. Используя одну функцию, переписать нечётные элементы целочисленных массивов M[10] и P[16] в массив B подряд. Вывести на печать исходные и полученные массивы.

Вариант 12. Определить функцию, которая заменяет элементы целочисленного массива, меньшие заданной величины V, на квадрат своих значений. Использовать функцию для изменения массивов A1[13] и A2[15].

Вариант 13. Используя одну функцию, вычислить произведение и количество ненулевых элементов в целочисленных массивах F[12] и P[15].

Вариант 14. Описать функцию, которая возвращает индекс минимального элемента в числовом массиве из N элементов. Определить, в каком из массивов L[15] или V[20], минимальный элемент встречается раньше.

Вариант 15. Используя одну функцию, определить число элементов до первого отрицательного в целочисленных массивах A[10], B[15], C[20] и сформировать из таких элементов массив F.

Вариант 16. Описать функцию, которая возвращает 1, если числовой массив из N элементов упорядочен по возрастанию, и 0 – в противном случае. Для массивов T[15], K[30] и L[20] вывести сообщение о том, упорядочены они или нет.

Вариант 17. Описать функцию, которая заменяет все элементы со значением d на значения r. В массиве Q[30] заменить нулевые элементы на -1, а в массиве S[15] заменить значения 10 на 100.

Вариант 18. Определить функцию определения среднего арифметического в массиве из N элементов. Вычислить Ucp + Tcp +Gcp для массивов U[10], T[12] и G[15].

Вариант 19. Используя одну функцию, сформировать одномерный массив S значениями 10, 20, 30, …, 100, массив T – значениями 3, 6, 9, … 30, массив D – значениями 5, 10, 15, … 50.

Вариант 20. Используя одну функцию, определить среднее арифметическое и среднее геометрическое в целочисленных массивах mmm[10] и ppp[12].

Пример выполнения практического задания №3.

Задание.

Используя одну функцию, определить номер минимального элемента в одномерном вещественном массиве и количество элементов, стоящих после него в массивах W[22] и V[25].

Решение.

1) Словесный алгоритм.

Для заполнения и вывода массивов используются функции: FillArray и OutputArray. Далее в теле программы main вызываются функции заполнения и вывода массива для массивов W и V. Функция, вычисляющая номер минимального элемента и количество элементов после него - Calc. Данная функция принимает в качестве параметра массив из целых числе, равный длине передаваемого массива, а также ссылки на переменные count и number. Переменная count отвечает за количество элементов в массиве, находящихся после минимального, а переменная number – за номер минимального элемента в массиве.

2) Схема алгоритма. Схема алгоритма программы представлена на рисунке 11-14.

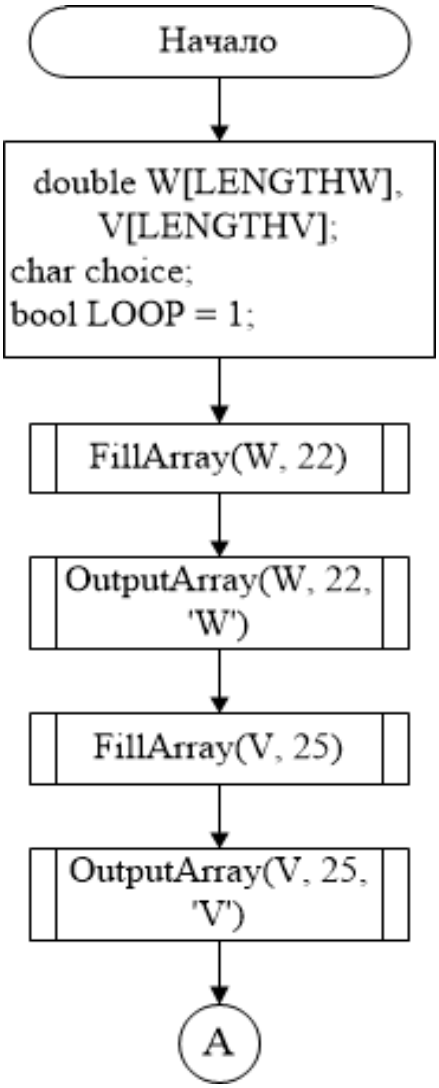

Рисунок 11, Лист 1 - Схема алгоритма реализации алгоритма работы с применением пользовательских функций

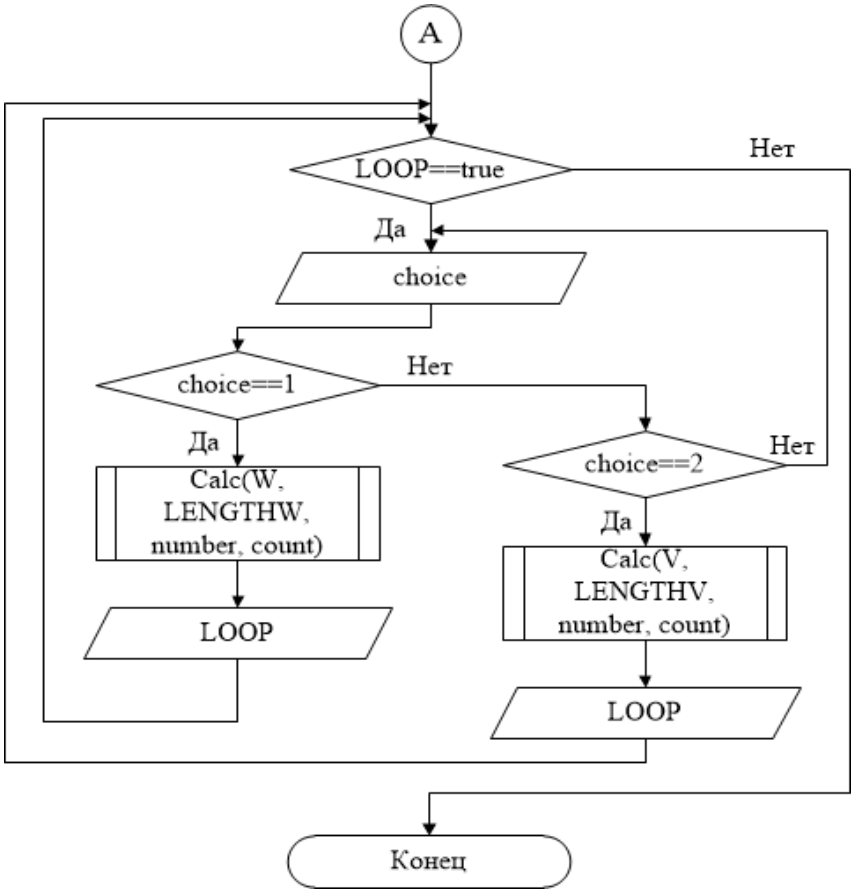

Рисунок 11, Лист 2

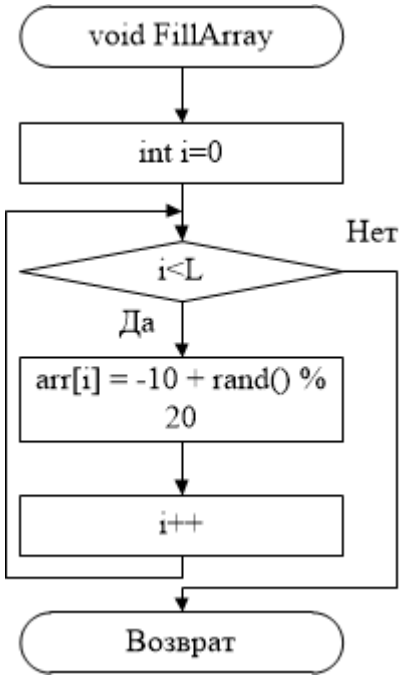

Рисунок 12 – Схема алгоритма функции FillArray

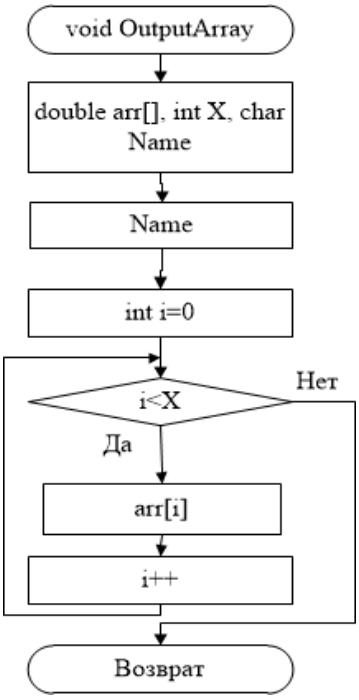

Рисунок 13 – Схема алгоритма функции OutputArray

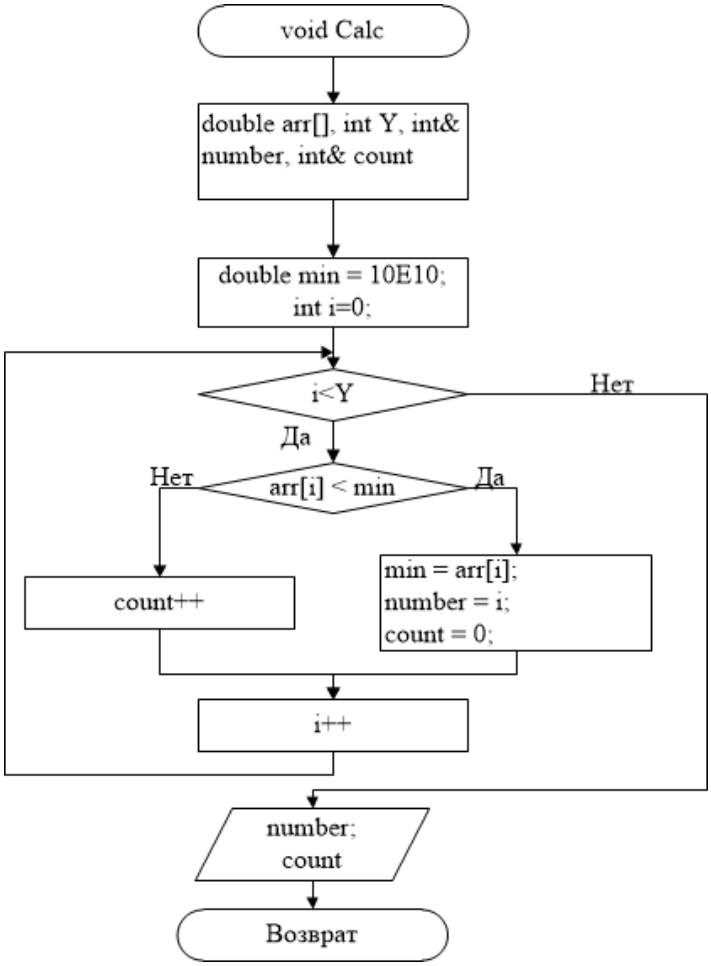

Рисунок 14 – Схема алгоритма функции calc

3) Код программы.

```
#include "pch.h"
#include <iostream>
#include <ctime>
using namespace std;
void FillArray ( double arr[], int L)                    // Заполнение массива
{
        for (int i = 0; i < L; i++)
        {       arr[i] = -10 + rand() % 20;     }
}
void OutputArray(double arr[], int X, char Name)              // Вывод массива
{
        cout << "Массив "<<Name<<": " << endl;
        for (int i = 0; i < X; i++)
        {
                cout << arr[i] << " ";
        }
        cout << endl; }
void Calc(double arr[], int Y, int& number, int& count)
{
        double min = 10E10;
        for (int i = 0; i < Y; i++)
        {
if (arr[i] < min)    // Если рассматриваемый элемент меньше min
                {
                        min = arr[i];               // Значение min меняется
                        number = i;                 // Сохраняется номер элемента
                        count = 0;                  // А счетчик элементов обнуляется
                }
```

```
            else count++;       // Иначе счетчик элементов увеличивается
            }
            cout << "number = " << number << endl;
            cout << "count = " << count<<endl; }
        int main()
        {
            setlocale(LC_ALL, "Rus");          // Установка русского языка в
консоли
            srand(time(NULL));
            const int LENGTHW = 22;
            const int LENGTHV = 25;
            double W[LENGTHW], V[LENGTHV];
            int number = 0, count = 0;
            char choice;
            bool LOOP=1;
            FillArray(W, 22);                  // Заполнение массива W
            OutputArray(W, 22, 'W');
            FillArray(V, 25);                  // Заполнение массива V
            OutputArray(V, 25, 'V');
            while (LOOP)
            {
                cout << "Выберите массив: \n1.W \n2.V"<<endl;
                cin >> choice;
                switch (choice)
                {
                case '1': { Calc(W, LENGTHW, number, count); cout <<
"Повторить?\(1/0)" << endl; cin >> LOOP; } break;       // Вызов функции для
массива W
```

case '2': { Calc(V, LENGTHV, number, count); cout << "Повторить?\(1/0)" << endl; cin >> LOOP; } break; // Вызов функции для массива V

default: cout << "Ошибка: Введено неверное значение. "; break;

}

}

4) Реализация программного кода. Реализация программного кода представлена на рисунке 15.

```
Массив W:
6 -4 1 -5 -5 -3 -3 1 -7 0 9 9 7 -7 -6 2 -6 8 -9 7 3 -10
Массив V:
2 3 0 9 -6 -3 3 1 0 -3 9 -10 -2 3 1 -5 7 0 -9 -4 8 -8 6 -5 -10
Выберите массив:
1.W
2.V
1
number = 21
count = 0
Повторить?(1/0)
1
Выберите массив:
1.W
2.V
2
number = 11
count = 13
Повторить?(1/0)
```

Рисунок 15 – Результат работы программы с использованием пользовательских функций для работы с массивами

4 Практическое занятие №4. Использование возможностей языка C++ для работы с файлами

Цель: Закрепление теоретических знаний и получение практических навыков обработки данных их файлов с помощью инструментов языка C++.

Задание: Составить программу, реализующую работу с файлами с помощью стандартных инструментов языка C++.

Вариант 1. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;
2) определять количество предложений в тексте;
3) по нажатию произвольной клавиши поочередно выделять каждое предложение текста.

Вариант 2. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;
2) определять количество слов в тексте;
3) по нажатию произвольной клавиши поочередно выделять каждое слово текста.

Вариант 3. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) определять количество слов в тексте, оканчивающихся на гласную букву;

3) по нажатию произвольной клавиши поочередно выделять каждое слово текста, оканчивающееся на гласную букву.

Вариант 4. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) определять количество слов в тексте, у которых первый и последний символы совпадают;

3) по нажатию произвольной клавиши поочередно выделять каждое слово текста в последовательности 2, 3, 1.

Вариант 5. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) определять количество слов в тексте, у которых первый и последний символы совпадают;

3) по нажатию произвольной клавиши поочередно выделять каждое слово текста, у которого первый и последний символы совпадают.

Вариант 6. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) определять количество слов в тексте, начинающихся на гласную букву;

3) по нажатию произвольной клавиши поочередно выделять каждое предложение текста, начинающееся на гласную букву.

Вариант 7. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

4) вывод текста на экран;

5) определять количество символов в самом длинном слове;

6) по нажатию произвольной клавиши поочередно выделять каждое предложение текста, содержащее максимальное количество символов.

Вариант 8. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) определять количество символов в самом коротком слове;

3) по нажатию произвольной клавиши поочередно выделять каждое предложение текста, а затем содержащее минимальное количество символов.

Вариант 9. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

7) вывод текста на экран;

8) определять количество предложений текста и количество символов, отличных от букв и пробела;

9) по нажатию произвольной клавиши поочередно выделять каждое предложение текста, а затем поочередно все символы, отличные от букв и пробела в выделенном предложении.

Вариант 10. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) определять количество предложений текста и количество слов в каждом предложении;

3) по нажатию произвольной клавиши поочередно выделять каждое предложение текста, а затем поочередно все слова в выделенном предложении.

Вариант 11. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) определять количество букв 'o' в последнем слове текста;

3) по нажатию произвольной клавиши поочередно выделять последнее слово текста, а в выделенном слове поочередно все буквы 'o'.

Вариант 12. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

10) вывод текста на экран;

11) определять самую длинную последовательность цифр в тексте (считать, что любое количество пробелов между двумя цифрами не прерывает последовательности цифр);

12) по нажатию произвольной клавиши поочередно выделять каждое последовательность цифр, содержащую максимальное количество символов.

Вариант 13. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 700 символов и длиной строки текста не

более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) определять порядковый номер заданного слова в каждом предложении текста (заданное слово вводится с клавиатуры);

3) по нажатию произвольной клавиши поочередно выделять каждое предложение текста, а затем заданное слово в выделенном предложении.

Вариант 14. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 700 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) выводить текст на экран дисплея еще раз, убирая из него заданное слово и удаляя лишние пробелы;

3) по нажатию произвольной клавиши поочередно выделять в тексте заданное слово (заданное слово вводить с клавиатуры).

Вариант 15. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 700 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) выводить текст на экран дисплея еще раз, меняя в нем местами заданные слова и удаляя лишние пробелы;

3) по нажатию произвольной клавиши поочередно выделять в тексте заданные слова, которые нужно поменять местами (заданные слова вводить с клавиатуры).

Вариант 16. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 700 символов и длиной строки текста не

более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) выводить текст на экран дисплея еще раз, заключая заданное слово в кавычки, и поочередно выделяет заданное слово вместе с кавычками;

3) по нажатию произвольной клавиши поочередно выделять в тексте заданное слово (заданное слово вводить с клавиатуры).

Вариант 17. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 700 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) выводить текст на экран дисплея еще раз, вставляя в каждое предложение в качестве последнего заданное слово, введенное с клавиатуры в качестве исходных данных;

3) по нажатию произвольной клавиши поочередно выделять в тексте вставленное слово.

Вариант 18. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 500 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) по нажатию произвольной клавиши поочередно выделять в тексте лишние пробелы между словами;

3) выводить текст на экран еще раз, убирая лишние пробелы между словами и начиная каждое предложение с новой строки.

Вариант 19. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 700 символов и длиной строки текста не

более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) по нажатию произвольной клавиши поочередно выделять в тексте заданное слово (заданное слово вводить с клавиатуры);

3) выводить текст на экран еще раз, заменяя в заданном слове строчные буквы прописными.

Вариант 20. Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) определить наибольшее количество подряд идущих пробелов в тексте;

3) по нажатию произвольной клавиши поочередно выделять каждую из последовательностей пробелов максимальной длины.

Пример выполнения практического задания №4.

Задание.

Необходимо используя текстовый редактор создать файл. В файл добавить текст, длиной не более 1000 символов и длиной строки текста не более 70 символов. Файл должен иметь расширение TXT. Разработать приложение, в котором предусмотреть решение следующих задач:

1) вывод текста на экран;

2) определить количество слов в тексте.

Решение.

1) Словесный алгоритм.

Для работы с файлами была подключена библиотека fstream. Затем для чтения данных из фала был создан объект file класса ifstream. С помощью метода

file.open был открыт файл text.txt находящийся в папке с проектом. Для вывода данных на экран используется функция getline. Для подсчета слов в цикле в переменную str вводилась строка до первого пробела и счетчик слов увеличивался на единицу. Затем количество символов выводится на экран.

2) Схема алгоритма. Схема алгоритма программы представлена на рисунке 15.

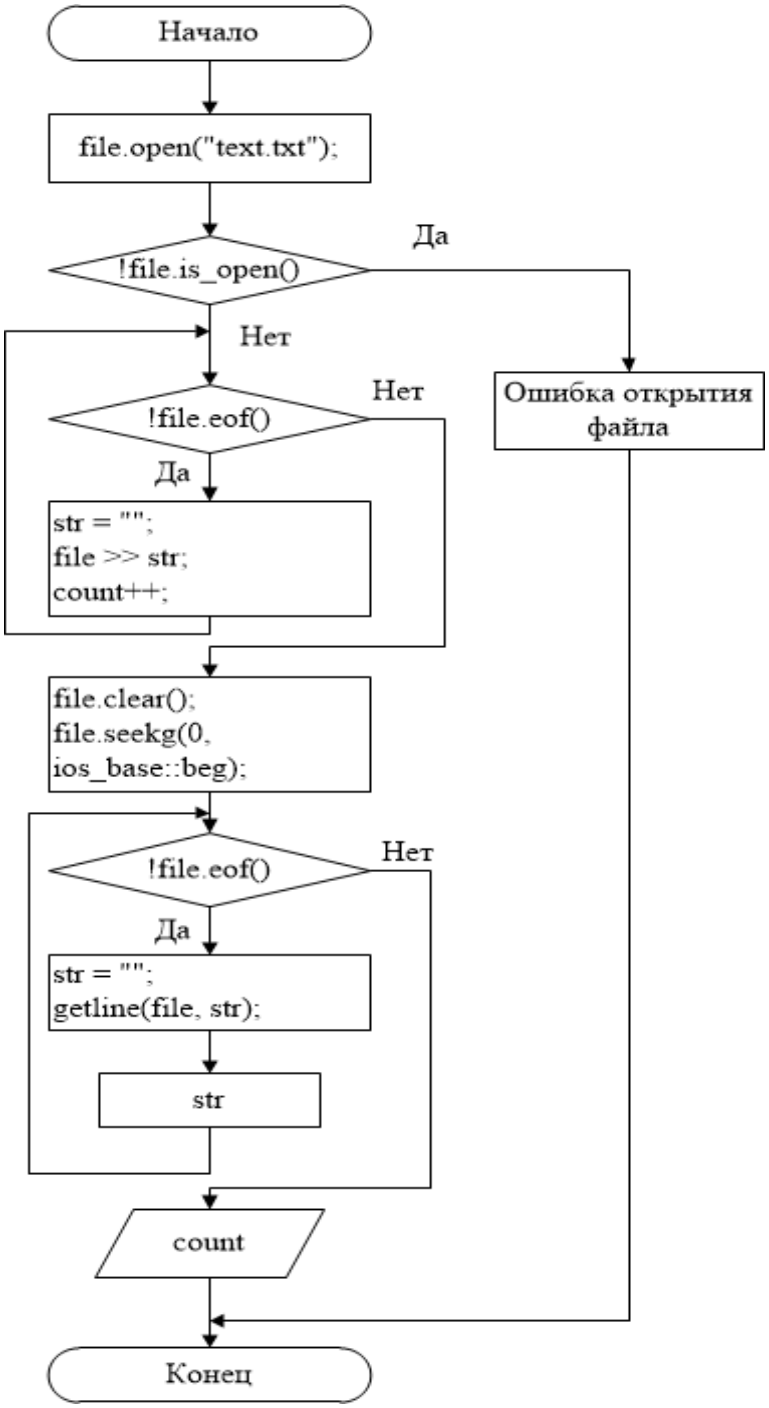

Рисунок 16 - Схема алгоритма реализации работы с файлами

3) Код программы.

```
#include "pch.h"
#include <iostream>
#include <string>
#include <fstream>
using namespace std;
int main()
{
	setlocale(LC_ALL, "Rus");
	ifstream file;
	file.open("text.txt");
	string str;
	int count = 0;
	if (!file.is_open())
	{
		cout << "Ошибка открытия файла";    }
	else
	{
		while (!file.eof())                    // Подсчет слов в файле
		{
			str = "";
			file >> str;
			count++;            }
		file.clear();
		file.seekg(0, ios_base::beg);          // Переход в начало файла
		while (!file.eof())
		{
			str = "";
			getline(file, str);
```

```
cout << str << endl;                // Вывод текста на дисплей
                }
        }
cout << "count = " << count << endl; // Вывод количество слов в файле
        file.close("text.txt"); }
```

4) Реализация программного кода. Реализация программного кода представлена на рисунках 17 и 18.

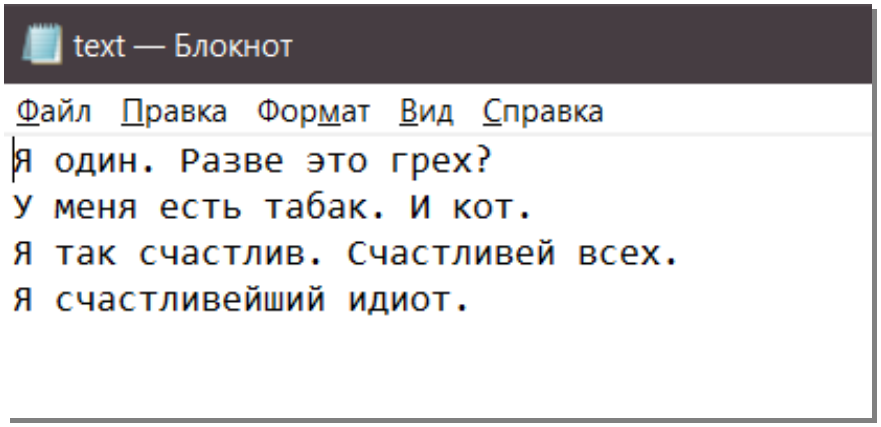

Рисунок 17 – Исходный файл

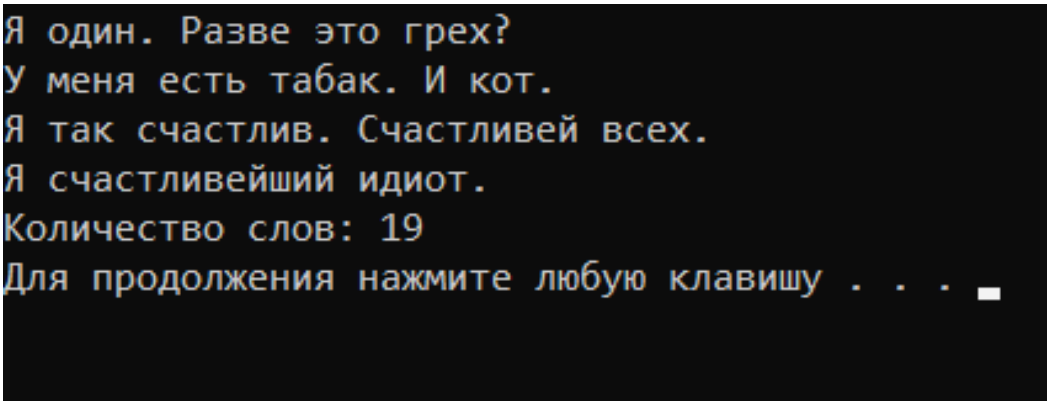

Рисунок 18 - Результат работы с файлами. Вывод количество слов

5 Практическое занятие №5. Создание приложений с использованием графических интерфейсов

Цель: Закрепление теоретических знаний и получение практических навыков визуального программирования на языке C++.

Задание: Составить программу с и использованием графических интерфейса пользователя. При работе приложения реализовать ввод и вывод данных через форму.

Вариант 1.Написать программу, основная форма которого имеет кнопку, открывающую новую форму. На новой форме должно быть две кнопки, первая открывает новую форму, а вторая закрывает эту форму и т.д. до пятой формы, которая имеет только одну кнопку «Закрыть».

Вариант 2. Написать программу, в которой создать голубое прямоугольное поле в нижней части окна, и красное поле такого же размера с цифрой 2. Требуется при помощи мыши перетаскивать красное поле в голубое. Если при отпускании левой кнопки мыши красное поле попадает в голубое, оно там фиксируется. В противном случае оно возвращается к своему исходному месту.

Вариант 3. Написать программу, в которой объявить двумерный массив и заполнить его построчно с клавиатуры. После заполнения – вывести заполненный массив на экран и посчитать сумму элементов отдельно в каждом столбце и каждой строке.

Вариант 4. Написать программу, в которой необходимо заполнить двумерный массив случайными числами от 10 до 100. После заполнения – вывести заполненный массив на экран и рассчитать сумму элементов отдельно в каждой строке и определить номер строки, в которой эта сумма максимальна.

Вариант 5. Написать программу, в которой пользователь вводит строку, состоящую из слов, разделенных пробелами. Требуется посчитать количество слов

в ней. Должно быть представлено два текстовых поля, в одно пользователь вводит строку, а в другом выводится результат обработки программой.

Вариант 6. Написать программу, в которой необходимо заполнить линейный массив случайными числами от –50 до 70, После заполнения – вывести заполненный массив на экран и рассчитать произведения наибольшего и наименьшего элементов массива.

Вариант 7. Написать программу, которая выводит введенное предложение, а затем, удаляет из него слова, которые состоят менее чем из трех букв. Должно быть представлено два текстовых поля, в одно пользователь вводит строку, а в другом выводится результат обработки программой.

Вариант 8. Написать программу, в которой пользователь вводит строку. Необходимо удалить из нее повторяющиеся символы и все пробелы. Должно быть представлено два текстовых поля, в одно пользователь вводит строку, а в другом выводится результат обработки программой.

Вариант 9. Написать программу, в которой пользователь вводит каждое слово построчно, а затем, программа сортирует слова в алфавитном порядке. Должно быть представлено текстовое поле, в которое пользователь вводит слова и поле вывода результата.

Вариант 10. Написать программу, в которой пользователь вводит строку, состоящую из букв и пробелов. Составить из входящих в нее букв несколько любых их сочетаний (слов) любой длины. Каждую букву строки можно использовать неограниченное количество раз. Должно быть представлено текстовое поле, в которое пользователь вводит слова и поле вывода результата.

Вариант 11. Написать программу, в которой необходимо заполнить два двумерных массива случайными числами от 10 до 100, После заполнения – вывести заполненные массивы на экран и перемножить их.

Вариант 12. Написать программу «Кошки», которая должна выводить информацию о заданной породе. Должно быть представлено текстовое поле, в

которое пользователь вводит интересующую его пароду кошки. Программа выполняет поиск по введенной пароде и выводит всю информацию о ней на экран.

Вариант 13. Написать программу «Планеты», которая должна выводить информацию о заданной планете. Должно быть представлено текстовое поле, в которое пользователь вводит интересующую его планету. Программа выполняет поиск по введенной планете и выводит всю информацию о ней на экран.

Вариант 14. Написать программу, обрабатывающую введенные пользователем строки. На форме должно быть реализовано два текстовых поля, в которые пользователь вводит строки для работы с ними. Необходимо реализовать обработку строк, используя следующее форматирование: реализовывать замену одной подстроки на другую.

Вариант 15. Написать программу позволяющую найти в заданном интервале автоморфные числа. Реализовать проверку введенного интервала: интервал не должен превышать определенное количество символов, значения интервала должны быть только цифрами и положительными. При ошибке выводить окно с пояснением ошибки.

Вариант 16. Написать программу «Справочник городов», который должен выводить информацию о заданном городе конкретной страны. Должно быть представлено текстовое поле, в которое пользователь вводит интересующий его город. Программа выполняет поиск по введенному городу и выводит всю информацию о нем на экран.

Вариант 17. Написать программу «Справочник рейсов», которая должна выводить о заданном номере рейса всю необходимую информацию. Должно быть представлено поле, в которое пользователь вводит интересующий его номер рейса. Программа выполняет поиск по введенному номеру рейса и выводит всю информацию о нем на экран.

Вариант 18. Написать программу, обрабатывающую введенную пользователем строку. Должно быть реализовано текстовое поле, в которое пользователь вводит строку для работы с ней. Необходимо реализовать обработку

строк, используя следующее форматирование: подсчитать, сколько во введенной строке заданных букв;

Вариант 19. Написать программу, обрабатывающую введенную пользователем строку. Должно быть реализовано текстовое поле, в которое пользователь вводит строку для работы с ней. Необходимо реализовать обработку строк, используя следующее форматирование: определить длину строки посимвольно.

Вариант 20. Написать программу, обрабатывающую введенную пользователем строку. На форме должно быть реализовано текстовое поле, в которое пользователь вводит строку для работы с ней. Необходимо реализовать обработку строк, используя следующее форматирование: разбить строку на подстроки.

Пример выполнения практического задания №5.

Задание.

Написать программу, обрабатывающую введенную пользователем строку. На форме должно быть реализовано текстовое поле, в которое пользователь вводит строку, состоящую из букв и пробелов. Необходимо реализовать обработку строк, используя следующее форматирование: посчитать число пробелов в введенной строке.

Решение.

1) Словесный алгоритм.

При разработке программы в диалоговом окне были созданы два управляющих поля: одно для ввода строки, другое для вывода результата. В поле ввода строки добавлена переменная типа CString с именем str. В поле вывода результата добавлена переменная типа int с именем count. При нажатии на кнопку «Выполнить» программа подсчитывает количество пробелов во веденной строке. Для получения введенных данных используется функция UpdateData(TRUE). Для

поиска пробелов в строке используется функция isspace(str[i]) которая возвращала значение TRUE если рассматриваемый символ оказывался пробелом. Для отображения данных в поля используется функция UpdateData(FALSE).

2) Схема алгоритма. Схема алгоритма программы представлена на рисунках 18-19.

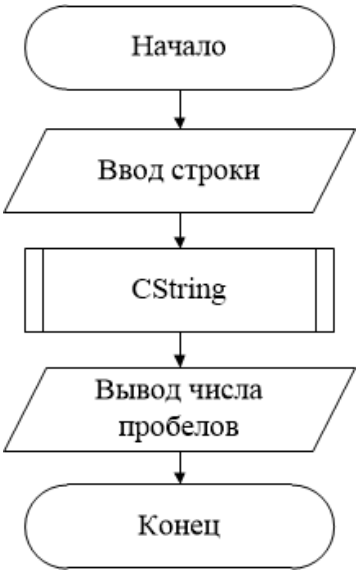

Рисунок 19 – Схема алгоритма использования графического интерфейса в приложении

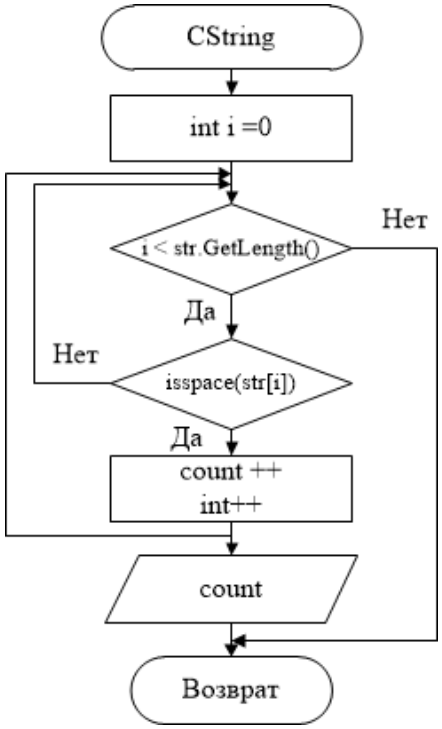

Рисунок 20 - Схема реализации алгоритма поиска пробелов в строке

3) Код программы.

```
#include "stdafx.h"
#include "MFCApplication10.h"
#include "MFCApplication10Dlg.h"
#include "afxdialogex.h"
#include "string.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
// Диалоговое окно CAboutDlg используется для описания сведений о приложении
class CAboutDlg : public CDialogEx
{
public:
	CAboutDlg();
// Данные диалогового окна
#ifdef AFX_DESIGN_TIME
	enum { IDD = IDD_ABOUTBOX };
#endif
	protected:
	virtual void DoDataExchange(CDataExchange* pDX);    // поддержка DDX/DDV
// Реализация
protected:
	DECLARE_MESSAGE_MAP()
};
CAboutDlg::CAboutDlg() : CDialogEx(IDD_ABOUTBOX)
{
}
```

```
void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialogEx::DoDataExchange(pDX);
}
BEGIN_MESSAGE_MAP(CAboutDlg, CDialogEx)
END_MESSAGE_MAP()
// Диалоговое окно CMFCApplication10Dlg
CMFCApplication10Dlg::CMFCApplication10Dlg(CWnd* pParent /*=nullptr*/)
	: CDialogEx(IDD_MFCAPPLICATION10_DIALOG, pParent)
	, str(_T(""))
	, count(0)
{
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}
void CMFCApplication10Dlg::DoDataExchange(CDataExchange* pDX)
{
	CDialogEx::DoDataExchange(pDX);
	DDX_Text(pDX, IDC_EDIT1, str);
	DDX_Text(pDX, IDC_EDIT2, count); }
BEGIN_MESSAGE_MAP(CMFCApplication10Dlg, CDialogEx)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_BN_CLICKED(IDC_BUTTON1, &CMFCApplication10Dlg::OnBnClickedButton1)
END_MESSAGE_MAP()
// Обработчики сообщений CMFCApplication10Dlg
BOOL CMFCApplication10Dlg::OnInitDialog()
```

```
{
    CDialogEx::OnInitDialog();
    // Добавление пункта "О программе..." в системное меню.
    // IDM_ABOUTBOX должен быть в пределах системной команды.
    ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
    ASSERT(IDM_ABOUTBOX < 0xF000);
    CMenu* pSysMenu = GetSystemMenu(FALSE);
    if (pSysMenu != nullptr)
    {
        BOOL bNameValid;
        CString strAboutMenu;
        bNameValid = strAboutMenu.LoadString(IDS_ABOUTBOX);
        ASSERT(bNameValid);
        if (!strAboutMenu.IsEmpty())
        {
            pSysMenu->AppendMenu(MF_SEPARATOR);
            pSysMenu->AppendMenu(MF_STRING,
IDM_ABOUTBOX, strAboutMenu);          }
    }
    // Задает значок для этого диалогового окна.  Среда делает это автоматически, если главное окно приложения не является диалоговым
    SetIcon(m_hIcon, TRUE);              // Крупный значок
    SetIcon(m_hIcon, FALSE);          // Мелкий значок
    // TODO: добавьте дополнительную инициализацию
    return TRUE;  // возврат значения TRUE, если фокус не передан элементу управления
}
void CMFCApplication10Dlg::OnSysCommand(UINT nID, LPARAM lParam)
```

```
{
    if ((nID & 0xFFF0) == IDM_ABOUTBOX)
    {
        CAboutDlg dlgAbout;
        dlgAbout.DoModal();   }
    else
    {
        CDialogEx::OnSysCommand(nID, lParam);    }
}
```

// При добавлении кнопки свертывания в диалоговое окно нужно воспользоваться приведенным ниже кодом, чтобы нарисовать значок. Для приложений MFC, использующих модель документов или представлений, это автоматически выполняется рабочей областью.

```
void CMFCApplication10Dlg::OnPaint()
{
    if (IsIconic())
    {
        CPaintDC dc(this); // контекст устройства для рисования
        SendMessage(WM_ICONERASEBKGND,
reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);
        // Выравнивание значка по центру клиентского прямоугольника
        int cxIcon = GetSystemMetrics(SM_CXICON);
        int cyIcon = GetSystemMetrics(SM_CYICON);
        CRect rect;
        GetClientRect(&rect);
        int x = (rect.Width() - cxIcon + 1) / 2;
        int y = (rect.Height() - cyIcon + 1) / 2;
        // Нарисуйте значок
```

```
            dc.DrawIcon(x, y, m_hIcon); }
        else
        {
            CDialogEx::OnPaint();        }
    }
    // Система вызывает эту функцию для получения отображения курсора при перемещении свернутого окна.
    HCURSOR CMFCApplication10Dlg::OnQueryDragIcon()
    {
        return static_cast<HCURSOR>(m_hIcon);
    }
    void CMFCApplication10Dlg::OnBnClickedButton1()  // Обработчик нажатия на кнопку «Выполнить»
    {   UpdateData(TRUE);
        for (int i = 0; i < str.GetLength(); i++)
        {           if (isspace(str[i])) count++;    }
        UpdateData(FALSE); }
```

4) Реализации программного кода. Реализация программного кода представлена на рисунке 21.

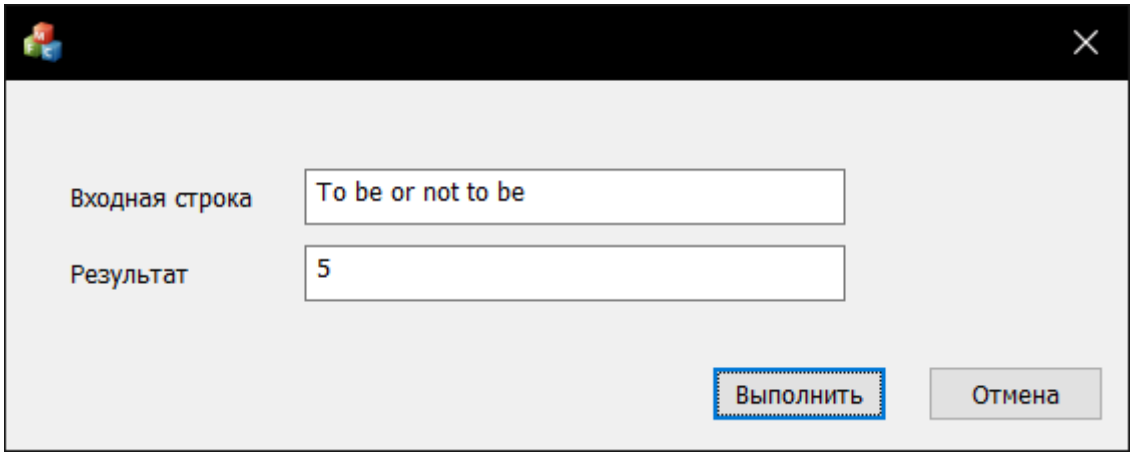

Рисунок 21 - Результат работы программы с графическим интерфейсом. Поиск пробелов в строке

6 Практическое занятие №6. Использование элементов объектно-ориентированного программирования при разработке приложений на языке C++

Цель: Закрепление теоретических знаний и получение практических навыков работы с классами в C++, использование в программе массива объектов. Изучение наследования в классах, виртуальных методов.

Задание: Разработать программу на языке C++ с использованием элементов объектно-ориентированного программирования.

Вариант 1. Необходимо описать класс, реализующий стек. Необходимо разработать приложение, использующее этот класс для моделирования Т-образного сортировочного узла на железной дороге. Должна быть реализована возможность разделять на два направления состав, состоящий из вагонов двух типов (на каждое направление формируется состав из вагонов одного типа).

Необходимо разработать приложение показывающее работу с классом А также необходимо предусмотреть возможность формирования состава из файла и с клавиатуры.

Вариант 2. Необходимо описать класс, реализующий бинарное дерево, который должен обладать возможностью добавления новых элементов, удаления существующих, поиска элемента по ключу, а также последовательного доступа ко всем элементам.

Необходимо разработать приложение показывающее работу с классом для представления англо-русского словаря. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса. А также необходимо предусмотреть возможность формирования словаря из файла и с клавиатуры.

Вариант 3. Необходимо построить систему классов для описания плоских геометрических фигур: круг, квадрат, прямоугольник. Требуется предусмотреть следующие методы:

1) для создания объектов,
2) перемещения на плоскости,
3) изменения размеров и вращения на заданный угол.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 4. Необходимо построить описание класса, в котором должна содержаться информация об адресе предприятия. Требуется предусмотреть следующие операции: раздельное изменение составных частей адреса, создание и уничтожение объектов этого класса.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 5. Необходимо составить описание класса для представления комплексных чисел. Требуется предусмотреть следующие операции: сложения, вычитания, а также умножения комплексных чисел.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 6. Необходимо составить описание класса для объектов-векторов, которые должны задаваться координатами концов в трехмерном пространстве. Требуется предусмотреть следующие операции: сложения и вычитания векторов с получением нового вектора (суммы или разности), вычисления скалярного произведения двух векторов, длины вектора, а также косинуса угла между векторами.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 7. Необходимо составить описание класса прямоугольников со сторонами и параллельными осям координат. В программе должна быть возможность перемещения прямоугольников на плоскости, изменение размеров, построение наименьшего прямоугольника, а также содержащего два заданных прямоугольника, и прямоугольника, являющегося общей частью (пересечением) двух прямоугольников.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 8. Необходимо составить описание класса для определения одномерных массивов целых чисел (векторов). В программе должна быть возможность обращения к отдельному элементу массива с контролем выхода за пределы массива, возможность задания произвольных границ индексов при создании объекта и выполнения операций поэлементного сложения и вычитания массивов с одинаковыми границами индексов, а также умножения и деления всех элементов массива на скаляр. Должен осуществляться вывод элемента массива по заданному индексу и всего массива на экран.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 9. Необходимо составить описание класса для определения одномерных массивов строк фиксированной длины. В программе должна быть возможность обращения к отдельным строкам массива по индексам, контроль выхода за пределы массива, а также выполнения операций поэлементного сцепления двух массивов с образованием нового массива, слияния двух массивов с

исключением повторяющихся элементов. Должен осуществляться вывод элемента массива по заданному индексу и всего массива на экран.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 10. Необходимо составить описание класса многочленов от одной переменной, который должен задаваться степенью многочлена и массивом коэффициентов. В программе должны быть предусмотрены следующие методы:

1) для вычисления значения многочлена для заданного аргумента,
2) операции сложения,
3) вычитания и умножения многочленов с получением нового объекта-многочлена.

Должен осуществляться вывод описания многочлена на экран.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 11. Необходимо составить описание класса одномерных массивов строк, в котором каждая строка должна задаваться длиной и указателем на выделенную для нее память. Необходимо реализовать возможность обращения к отдельным строкам массива по индексам, затем контроль выхода за пределы массивов, выполнения операций поэлементного сцепления двух массивов с образованием нового массива, а также слияния двух массивов с исключением повторяющихся элементов. Должен осуществляться вывод элемента массива и всего массива на экран.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 12. Необходимо составить описание класса, который должен обеспечивать представление матрицы произвольного размера и иметь возможность

изменять числа строк, а также столбцов. Должен осуществляться вывода на экран подматрицы любого размера и всей матрицы.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 13. Необходимо написать класс для эффективной работы со строками. Реализовать возможность форматирования и сравнивания строк. В строках требуется хранить числовые значения, а также извлекать их. Для этого необходимо реализовать:

1) перегруженные операторы присваивания и конкатенации;
2) операции сравнения и приведения типов;
3) преобразование в число любого типа;
4) форматный вывод строки.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 14. Необходимо описать класс «кошки». Разработать приложение и предусмотреть работу с произвольным числом кошек, поиск кошек по признаку: по цвет, по размеру, по породе, а также реализовать добавления кошек в список, удаления из списка и сортировку по разным полям.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 15. Необходимо описать класс «актеры». Разработать приложение и предусмотреть работу с произвольным числом записей, поиска записи по признаку: по дате рождения, по цвету волос, по стране рождения, а также реализовать добавления и удаления записей, сортировки по разным полям.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 16. Необходимо описать класс «группа собак». Разработать приложение и предусмотреть работу с переменным числом собак, поиска собаки по признаку: по породе, по цвету, по размеру, а также реализовать добавления и удаления записей, сортировки по разным полям.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 17. Необходимо описать класс «вещественная матрица», который реализует тип данных. Разработать приложение, в котором предусмотреть реализацию следующих операций над матрицами:

1) сложение, вычитание, умножение, деление (умножение и деление должны осуществляться на другую матрицу, а так же на число);
2) комбинированные операции присваивания (+=, –=, *=, /=);
3) операции сравнения на равенство и неравенство;
4) операции вычисления обратной и транспонированной матрицы, операцию возведения в степень;
5) методы вычисления детерминанта и нормы;
6) методы, реализующие проверку типа матрицы (квадратная, диагональная, нулевая, единичная, симметрическая, верхняя треугольная, нижняя треугольная);
7) операции ввода и вывода в стандартные потоки.

В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 18. Необходимо описать класс «множество», который должен осуществлять такие операции как добавление и удаление элемента, пересечение, объединение и разность множеств.

Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Вариант 19. Необходимо описать класс, реализующий стек. Разработать приложение, позволяющее с помощью этого класс отыскать проход по лабиринту.

Представить лабиринт необходимо в виде матрицы, которая должна состоять из квадратов. Квадрат имеет одно свойство: либо открыт, либо закрыт. В закрытый квадрат должен быть запрещен вход. В открытые квадраты вход возможен только со стороны, с угла вход запрещен. В матрице квадрат должен определяться координатами. В конце работы - после отыскания прохода программа должна выводить найденный путь в виде координат квадратов.

Вариант 20. Необходимо описать класс «предметный указатель». В этом классе компонента указателя должна содержать слово и номера страниц, где это слово встречается. К одному слову количество номеров страниц должно быть от 1 до 10. Требуется предусмотреть возможность формирования указателя с клавиатуры, а также из файла, затем вывода указателя, вывода номеров страниц для заданного слова и удаление элемента из указателя. Необходимо разработать приложение показывающее работу с классом. В программе должно быть реализовано меню, с помощью которого должна осуществляться проверка всех методов этого класса.

Пример выполнения практического задания №6.

Задание.

Необходимо описать класса для представления комплексных чисел. Обеспечить выполнение операций сложения, вычитания и умножения комплексных чисел. Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Решение.

1) Словесный алгоритм.

В начале программы задается тело класса "Complex" для работы с комплексными числами. Переменные, необходимые для хранения чисел, объявлялись до части Public и автоматически помещались в Privat часть. В Public части заданы функции, предназначенные для сложения - sum, вычитания - diff и умножения – multip. Класс содержит конструктор для создания переменных, функцию для форматированного вывода, сложения, вычитания, умножения. Каждая из этих функций (кроме вывода) принимает значения типа Complex (чтобы не передавать 2 переменные, а только одну).

2) Схема алгоритма. Схема алгоритма программы представлена на рисунке 22.

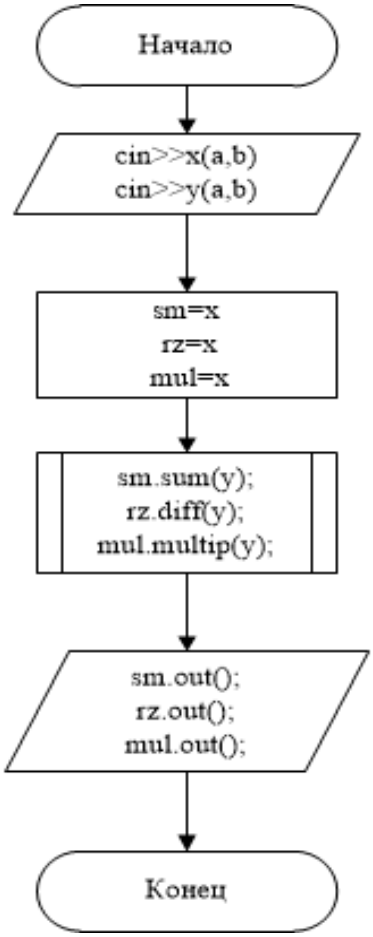

Рисунок 22 - Схема алгоритма программы с использованием элементов объектно-ориентированного программирования

3) Код программы.

```
#include <iostream>
using namespace std;
class Complex
{
	double a, b;
public:
	Complex() {
		a = 0;
		b = 0;		}
	Complex(double x, double y) {
		a = x;
		b = y;		}
	void out() {
		if (b < 0) {
			cout << "z1 = " << a << " - " << abs(b) << "i" << endl; }
		else cout << "z1 = " << a << " + " << b << "i" << endl; }
	void sum(Complex t) {// функция сложения чисел
		a += t.a;
		b += t.b;	}
	void diff(Complex t) {// функция разности
		a -= t.a;
		b -= t.b;	}
	void multip(Complex t) { // функция умножения
		a = a * t.a - b * t.b;
		b = a * t.a + b * t.b;		}
};
int main()
{
```

```
setlocale(LC_ALL, "Russian");
double a, b;
cout << "Введите два комплексных числа в формате a, b :" << endl;
cin >> a >> b;
Complex x(a, b);//создаем первое комплексное число
cin >> a >> b;
Complex y(a, b);//создаем второе комплексное число
Complex sm = x, rz = x, mul = x;//создаем буферные числа в которые будем класть результаты сложения, вычитания, умножения
system("cls");//отчищаем экран
cout << "Комплексные числа: " << endl;
x.out();//выводим  первое число
cout << endl;
y.out();// выводим второе число
cout << endl;
cout << "Сумма комплексных чисел: ";
sm.sum(y);//считаем сумму чисел
sm.out();
cout << endl;
cout << "Поизведение комплексных чисел: ";
mul.multip(y);//считаем произведение чисел
mul.out();
cout << endl;
cout << "Разность комплексных чисел: ";
rz.diff(y);//считаем разность чисел
rz.out();
cout << endl;
return 0; }
```

4) Реализация программного кода. Реализация программного кода представлена на рисунке 23.

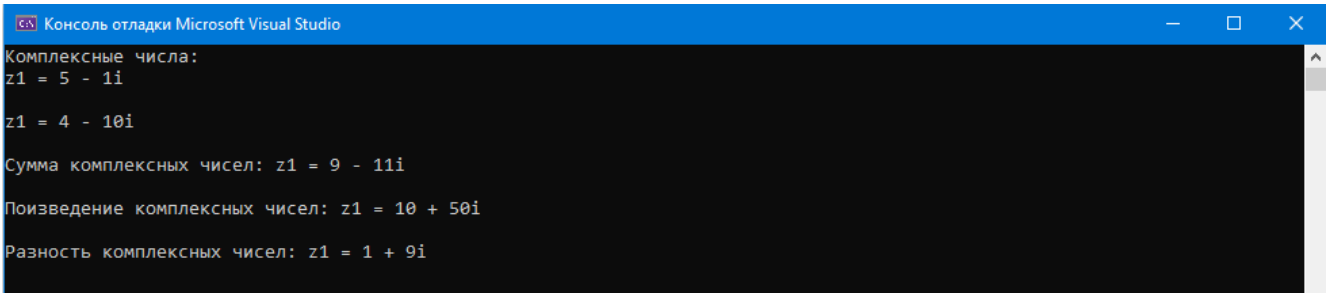

Рисунок 23 – Результат работы программы с использованием элементов объектно-ориентированного программирования

7 Практическое занятие №7. Создание приложений на языке C++ с использованием стандартных библиотек

Цель: Закрепление теоретических знаний и получения практических навыков использования библиотек языка C++.

Задание: Разработать программу согласно своему варианту с использованием необходимой библиотеки языка C++.

Вариант 1. Используя библиотеку math составить программу, вычисляющую значение функции: $f = \frac{cosa^5 * sinb^5}{\sqrt{ab}} - \frac{cosa^3 * sinb^3}{\sqrt{ab}}$.

Вариант 2. Используя библиотеку math составить программу, вычисляющую значение функции: $f = (arccosa)^5 * (sinb)^7 - (arcsina)^5 * (cosb)^7$.

Вариант 3. Используя библиотеку math составить программу, вычисляющую значение функции: $f = |\frac{cosa^4 * sinb^8}{arcsin\sqrt{ab}}|$.

Вариант 4. Используя библиотеку math составить программу, вычисляющую значение функции: $f = log\frac{cosa^3 * sinb^3}{\sqrt{sina * cosb}}$.

Вариант 5. Используя библиотеку math составить программу, вычисляющую значение функции: $f = \frac{cosa^e * sine^a}{cosa^4 * sine^6}$.

Вариант 6. Используя библиотеку cctype составить программу, которая удаляет все буквы из заданного пользователем массива.

Задание 7. Используя библиотеку cctype составить программу, которая удаляет все цифры из заданного пользователем массива.

Вариант 8. Используя библиотеку cctype составить программу, которая удаляет все буквы нижнего регистра из заданного пользователем массива.

Вариант 9. Используя библиотеку cctype составить программу, которая удаляет все буквы верхнего регистра из заданного пользователем массива.

Вариант 10. Используя библиотеку cctype составить программу, которая в заданном пользователем массиве заменяет все буквы верхнего регистра на буквы нижнего регистра.

Вариант 11. Используя библиотеку stdio составить программу, которая копирует данные из одного файла в другой, после чего удаляет первый файл.

Вариант 12. Используя библиотеку stdio составить программу, которая записывает в файл введенный пользователем текст. При вводе спецсимвола, печатает строку вида «s: error сообщение».

Вариант 13. Используя библиотеку stdio составить программу, выполняющую поиск и удаление введенного пользователем символа из файла.

Вариант 14. Используя библиотеку stdio составить программу, которая считывает текст из файла и выводит его на экран, после чего в конце каждой строки записывается цифра «1».

Вариант 15. Используя библиотеку stdio составить программу, которая перезаписывает информацию из первого файла во второй, а из второго файла в первый.

Вариант 16. Используя библиотеку string составить программу, которая выполняет поиск в массиве строк, и выводит строку, совпадающую с заданной.

Вариант 17. Используя библиотеку string составить программу, которая считывает строку и дописывает в ее конец числа от 1 до 10 в порядке возрастания.

Пример. Исходная строка: “Hello world”

“Hello world1”

“Hello world2”

…

“Hello world10”.

Вариант 18. Используя библиотеку string составить программу, которая считывает строку и с каждой последующей итерацией дописывает в ее начало числа от 1 до 10 с отступом после числа.

Пример. Исходная строка: “Hello world”

“1 Hello world”

“2 Hello world”

…

“10 Hello world”.

Вариант 19. Используя библиотеку string составить программу, которая считывает строку и с каждой последующей итерацией удаляет из нее последний символ, до окончательного удаления строки.

Пример. Исходная строка: “Hello world”

“Hello worl”

“Hello wor”

…

“H”

“”.

Вариант 20. Используя библиотеку string составить программу, которая выполняет замену заданного слова в массиве строк, на новое, вводимое пользователем.

Пример выполнения практического задания №7.

Задание.

Используя библиотеку string составить программу, которая считывает строку и дописывает в ее конец числа от 1 до 10 в порядке возрастания.

Решение.

1) Словесный алгоритм.

Сначала была задана строка типа string, далее она считывалась. В последующем цикле for создан указатель на массив типа char, а также строка для сохранения исходной строки неизменной. Далее номер итерации цикла переводится в строку и вставляется в конец локальной строки функцией append(). И измененная строка выводится на экран. Задана строка типа string, далее она считывалась. В последующем цикле for создан указатель на массив типа char, а также строка для сохранения исходной строки неизменной. Далее номер итерации цикла переводится в строку и вставляется в конец локальной строки функцией append(). Затем измененная строка выводится на экран.

2) Схема алгоритма. Схема алгоритма программы представлена на рисунке 23.

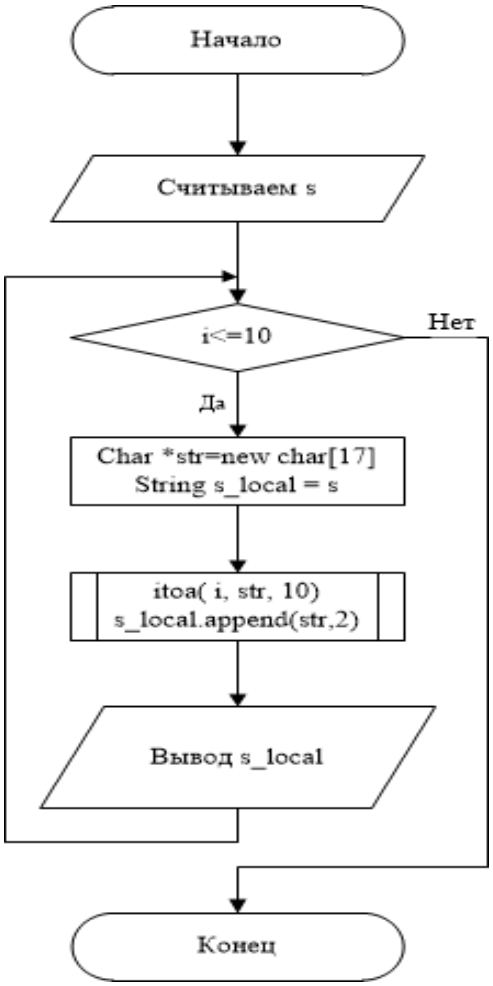

Рисунок 23 - Схема алгоритма программы с использованием библиотеки string

3) Код программы.

```
#include "pch.h"
#include <iostream>
#include <string>
using namespace std;
int main()
{       string s;
        getline(cin, s);// считываем строку
        for (int i = 1; i <= 10; i++)
        {
                char *str = new char[17];//массив для перевода I в строку
                string s_local;//локальная версия для изменения
                itoa(i, str, 10);//функция для перевода в ст1року
                s_local = s;
                s_local.append(str, 2);//добавление i в конец
                cout << s_local << endl;
                delete str;   }
        return 0; }
```

4) Реализация программного кода. Реализация программного кода представлена на рисунке 24.

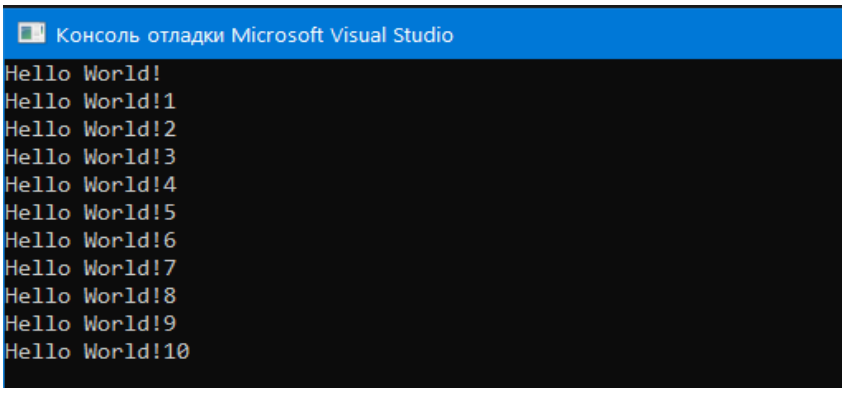

Рисунок 24 – Результат работы программы с использованием библиотеки string языка C++

Заключение

Таким образом, в представленных методических указаниях по практическим занятиям изложены задания для выполнения практических работ по разработке программ на языке C++. В данных методических указаниях по практическим работам представлена методика работы со структурами и объединениями, динамическими структурами данных, использование элементов объектно-ориентированного программирования. Помимо этого представлены задания по созданию собственных функций пользователя, созданию приложений с использованием графических интерфейсов, приложений на языке C++ с использованием стандартных библиотек.

Таким образом, в результате изучения представленных методических указаний студенты будут уметь самостоятельно разрабатывать приложения на языке C++.

Список использованных источников

1 Павловская, Т.А. С/С++. Программирование на языке высокого уровня / Т. А. Павловская. - СПб.: Питер, 2004. – 461 с.

2 Павловская, Т.А. С/С ++. Структурное программирование: Практикум / Т.А. Павловская, Ю.А. Щупак. - СПб.: Питер, 2007. - 239 с.: ил.

3 Павловская, Т. А., С++. Объектно-ориентированное программирование: Практикум. / Т.А. Павловская, Ю.А. Щупак - СПб.: Питер, 2006. - 265 с: ил.

4 Кольцов, Д.М. 100 примеров на Си. / Д.М. Кольцов. СПб.: “Наука и техника”, 2017 - 256 с.

5 Доусон, М. Изучаем С++ через программирование игр. / М. Доусон. - СПб.: “Питер”, 2016. - 352.

6 Седжвик, Р. Фундаментальные алгоритмы на С++. Анализ/Структуры данных/Сортировка/Поиск: Пер. с англ. / Роберт Седжвик. - Киев: Издательство “Диасофт”, 2001. - 688с.

7 Сиддкхартха, Р. Освой самостоятельно С++ за 21 день. / Сиддкхартха, Р. - М.: SAMS, 2013. - 651 с.

8 Стивен, П. Язык программирования С++. Лекции и упражнения: пер. с англ. / П. Стивен. 6-е изд. - М.: ООО "И.Д. Вильямс", 2012. - 1248 с.

9 Черносвитов, А. Visual С++: руководство по практическому изучению / А. Черносвитов . - СПб. : Питер, 2002. - 528 с. : ил.

10 Страуструп, Б. Дизайн и эволюция языка С++. / Б. Страуструп - М.: ДМК, 2000. - 448 с.

11 Мейерс, С. Эффективное использование С++. / С. Мейерс. - М.: ДМК, 2000. - 240 с.

12 Бадд Т. Объектно-ориентированное программирование в действии. - СПб: Питер, 1997. - 464 с.

13 Лаптев В.В. С ++. Объектно-ориентированное программирование: Учебное пособие.- СПб.: Питер, 2008. - 464 с.: ил.

14 Страуструп, Б. Язык программирования С++. / Б. Страуструп. Режим доступа: http://8361.ru/6sem/books/Straustrup-Yazyk_programmirovaniya_c.pdf.

15 Керниган, Б., Язык программирования Си. / Б. Керниган, Д. Ритчи. Режим доступа: http://cpp.com.ru/kr_cbook/index.html.

16 Герберт, Шилдт: С++ базовый курс. / Герберт Шилдт. Режим доступа: https://www.bsuir.by/m/12_100229_1_98220.pdf.

17 Богуславский, А.А. Основы программирования на языке Си++. / А.А. Богуславский, С.М. Соколов Режим доступа: http://www.ict.edu.ru/ft/004246/cpp_p1.pdf.

18 Линский, Е. Основы С++. / Е. Линский. Режим доступа: https://www.lektorium.tv/lecture/13373.

Приложение А

Структура отчета

Отчет по практическому заданию должен был выполнен на листах формата А4 и оформлен в соответствии с требованиями СТО.

Отчет по практическому заданию должен содержать следующие пункты:

1) Задание на практическое занятие.
2) Этапы решения задачи.
3) Словесный алгоритм.
4) Схема алгоритма.
5) Код программы.
6) Реализация программного кода.