## Титульный лист пояснительной записки к курсовой работы



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА—Российский технологический университет»**

**РТУ МИРЭА**

Институт кибербезопасности и цифровых технологий

(наименование института, филиала)

Кафедра КБ-2 «Прикладные информационные техологии»

(наименование кафедры)

**КУРСОВАЯ РАБОТА**

по дисциплине Языки программирования

(наименование дисциплины)

**Тема курсовой работы** Разработка программы информационного поиска студентов по заданным критериям с возможностью шифрования данных

Студент группы Мысливец Леонид Владимирович \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

учебная группа, фамилия, имя, отчество студента подпись студента

Руководитель курсовой работы Серов В. А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_

должность, звание, ученая степень, подпись руководителя

Рецензент (при наличии)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

должность, звание, ученая степень подпись рецензента

Работа представлена к защите «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

Допущен к защите «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_

|  |
| --- |
| Бланк задания на курсовую работу   МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА**  Институт кибербезопасности и цифровых технологий |
| Кафедра КБ-2 «Прикладные информационные технологии»  (наименование кафедры)  Утверждаю  Заведующий кафедрой КБ-2  Трубиенкао О.В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_202\_ г. |
| ЗАДАНИЕ |
| на выполнение курсовой работы по дисциплине  «Языки программирования» |

Студент: Мысливец Леонид Владимирович Группа: БББО-01-21

Тема: массив, структура, файл, функции, классы, объекты, инкапсуляция, полиморфизм и другие элементы ООП

**Исходные данные**: Информация о группе студентов из N человек, где запись о студенте содержит следующие данные: Фамилия, имя, отчество студента, Число, месяц, год рождения, Год поступления в институт, Факультет(институт), кафедра, Группа, Номер зачетной книжки, Пол, Названия предметов и оценки по каждому, предмету в каждой сессии, максимально 9 сессий и 10 предметов в каждом семестре, которые могут быть разные. Все данные должны быть форматными: даты, числа и т.д. Задание, варианта определяется 2-мя последними цифрами зачетной книжки.

**Перечень вопросов, подлежащих разработке, и обязательного графического материала:** массив, структура, файл, функции, классы, объекты, инкапсуляция, полиморфизм и другие элементы ООП.

Срок представления к защите курсового проекта (работы) до «15» мая 2022г.

Задание на выполнение курсовой работы выдал\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_)

ф.и.о. руководителя подпись руководителя «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2022г.

Задание на курсовую проект работу получил \_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

ф.и.о. исполнителя подпись обучающегося

СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 3](#_Toc103526009)

[ВВЕДЕНИЕ 5](#_Toc103526010)

[1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 8](#_Toc103526011)

[1.1 Массив символов 8](#_Toc103526012)

[1.2 Массивы указателей 8](#_Toc103526013)

[1.3 Ссылки 9](#_Toc103526014)

[1.4 Работа с памятью. Указатели 9](#_Toc103526015)

[1.5 Стандартные функции работы с файлами. 10](#_Toc103526016)

[1.5.1 Функция открытия файла 11](#_Toc103526017)

[1.5.2 Функция закрытия файла 11](#_Toc103526018)

[1.5.3 Функция контроля конца файла 11](#_Toc103526019)

[1.6 Динамические структуры данных 11](#_Toc103526020)

[1.7 Линейный список 12](#_Toc103526021)

[1.8 Парадигмы ООП – инкапсуляция, наследование, полиморфизм 12](#_Toc103526022)

[1.9 Использование динамических массивов. 12](#_Toc103526023)

[1.10 Классы. Программирование линейных алгоритмов с использованием функций инициализации set() 13](#_Toc103526024)

[1.11 Конструктор класса 13](#_Toc103526025)

[1.12 Деструктор класса 13](#_Toc103526026)

[1.13 Перегрузка операций 14](#_Toc103526027)

[1.14 Наследование классов 14](#_Toc103526028)

[2 ПРАКТИЧЕСКАЯ ЧАСТЬ 15](#_Toc103526029)

[2.1 Структура программы 15](#_Toc103526030)

[2.2 Алгоритм решения задачи 16](#_Toc103526031)

[Анализ результатов 19](#_Toc103526032)

[Листинг кода решения задачи 21](#_Toc103526033)

[ЗАКЛЮЧЕНИЕ 22](#_Toc103526034)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 23](#_Toc103526035)

[ПРИЛОЖЕНИЕ 24](#_Toc103526036)

# ВВЕДЕНИЕ

Курсовая работа является формой самостоятельной работы обучающихся под руководством преподавателя и представляет собой творческую, самостоятельную работу обучающихся, имеющую целью формирование у них компетенций с учетом требований Федеральных государственных образовательных стандартов высшего образования по направлению (специальности) подготовки.

Настоящие методические рекомендации по выполнению курсовой работы разработаны в соответствии с Федеральным законом от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации», приказом Министерства образования и науки Российской Федерации от 05 апреля 2017 г. № 301 «Об утверждении Порядка организации и осуществления образовательной деятельности по образовательным программам высшего образования - программам бакалавриата, программам специалитета, программам магистратуры». Уставом и локальными нормативными актами Университета.

Основной целью курсовой работы является формирование и закрепление компетенций путём практического использования знаний, умений и навыков, полученных в рамках теоретического обучения, а также выработка самостоятельного творческого подхода к решению конкретных профессиональных задач в области разработки прикладного программного обеспечения.

Курсовая работа выполняется лично студентом под руководством преподавателя.

Закрепление тем курсовой работы за обучающимися и назначение руководителей производится распоряжением заведующего кафедрой и обеспечивается соответствующим бланком задания на курсовую работу, утверждаемым заведующим кафедрой.

Список примерных (типовых) тем курсовой работы приведен в соответствующем разделе конкретной ООП (рабочей программе дисциплины «языки программирования»). Тематика курсовой работы по дисциплине «Языки программирования» актуализируется и утверждается на заседании кафедры КБ-2 «Прикладные информационные технологии». В данных методических рекомендациях приводится одна из таких типовых тем, а также варианты конкретного задания.

Перечень вариантов курсовой работы доводится до сведения обучающихся в течение первых трех недель семестра, в котором предусмотрено учебным планом выполнение курсовой работы. Выбор номера варианта курсовой работы носит случайный характер. Для упорядочивания порядка определения конкретного номера варианта используется следующий принцип: ***номер варианта курсовой работы есть младшие два числовых разряда номера зачетной книжки студента, выполняющего курсовую работу***.

Обучающийся имеет право выбора темы из списка, предложенного кафедрой, которое оформляется личным заявлением на имя заведующего кафедрой. Обучающийся может предложить свою тему при условии обоснования ее целесообразности. Темы курсовой работы обучающихся должны быть определены не позднее трех недель с начала соответствующего семестра.

По обоснованному решению кафедры данный срок может быть в виде исключения изменен.

Копии распоряжения заведующего кафедрой (выписки из протокола заседания кафедры) передаются в учебно-методический отдел института кибербезопасности и цифровых технологий для учета и внесения в информационно-аналитическую систему «Университет».

Решением кафедры допускается изменение темы курсовой работы по личному заявлению обучающегося, согласованному с руководителем и заведующим кафедрой, при этом оформляется новое задание на курсовую работу и издается соответствующее распоряжение заведующего кафедрой.

Лучшие курсовые работы представляющие учебно-методическую ценность, могут быть использованы в качестве учебных пособий (с указанием авторства), либо могут быть представлены на соответствующие конкурсы студенческих работ.

Практические работы, связанные с курсовой работой, выполняются с использованием персональных компьютеров. Указания по технике безопасности совпадают с требованиями, предъявляемыми к пользователю ЭВМ. Другие опасные факторы отсутствуют.

Вариант 46

# 1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

**1.1 Массив символов**

Символьная переменная (переменная типа char) — это величина размером в 1 байт, которая используется для представления литер и целых чисел в диапазоне от 0 до 255 или от -128 до 127, в зависимости от того, знаковая эта переменная или беззнаковая. Символьные константы заключаются в одинарные кавычки. Примеры символьных констант: 'a', '+', '1'.

Строковая константа, или строковый литерал — это нуль или более символов, заключенных в двойные кавычки, например,

Таким образом, в языке С строка (строковая константа) — это массив символов, заканчивающийся нулевым байтом. Поэтому памяти для строки требуется на один байт больше, чем число символов, расположенных между двойными кавычками.

Строковые константы размещаются в области данных. Вся фраза в кавычках является указателем на место в памяти, где записана строка. Это аналогично имени массива, служащего указателем на адрес расположения массива. Для вывода символьной константы служит идентификатор %s.

**1.2 Массивы указателей**

Массив указателей – простейшая структура данных, в которой проявляется различие между физическим и логическим порядком следования элементов. Способ организации данных ясен уже из самого определения: это массив, каждый элемент которого содержит указатель на переменную (объект).

Если это записать в терминах контекстного определения переменных, то получим, например

Переменную p следует понимать как массив (операция []), каждым элементом которого является указатель на переменную типа double (операция \*). Переменная p является массивом указателей как тип данных, но не является таковой как структура данных. Чтобы превратиться в структуру данных, она быть дополнена указуемыми переменными и указателями (связями).

Указатель на указатель на отдельную переменную, имеет довольно специфическое применение. Обычно от используется для передачи адреса заголовка какой-либо структуры данных при необходимости его изменения

**1.3 Ссылки**

Ссылка - это псевдоним для другой переменной. Они объявляются при помощи символа &. Ссылки должны быть проинициализированы при объявлении, причем только один раз.

Ссылка при определении сразу же инициализируется. Инициализация ссылки производится следующим образом:

int i = 0;

int& iref = i;

Физически iref представляет собой постоянный указатель на int - переменную типа int\* const.

Поскольку ссылка — это псевдоним, то при передаче объекта в функцию по ссылке внутри нее объект можно изменять. Ссылки не могут ссылаться на другие ссылки или на поле битов. Не может быть массивов ссылок или указателей на ссылку. Ссылка может использоваться для возврата результата из функции. Возвратить результат по ссылке - значит возвратить не указатель на объект и не его значение, а сам этот объект.

**1.4 Работа с памятью. Указатели**

Указатели являются одним из основных понятий языка Си. В такие переменные можно записывать адреса любых участков памяти, на чаще всего – адрес начального элемента динамического массива.

указатель – это переменная, в которой записан адрес другой переменной;

при объявлении указателя надо указать тип переменных, на которых он будет указывать, а перед именем поставить знак \*;

знак & перед именем переменной обозначает ее адрес;

знак \* перед указателем в рабочей части программы (не в объявлении) обозначает значение ячейки, на которую указывает указатель;

нельзя записывать по указателю, который указывает непонятно куда – это вызывает сбой программы, поскольку что-то стирается в памяти;

для обозначения недействительного указателя используется константа NULL;

при изменении значения указателя на n он в самом деле сдвигается к n-ому следующему числу данного типа, то есть для указателей на целые числа на n\*sizeof(int) байт;

указатель печатаются по формату %p.

Динамическое выделение памяти

Динамическими называются массивы, размер которых неизвестен на этапе написания программы. Прием, о котором будем говорить, относится уже не к стандартному языку Си, а к его расширению Си ++. Существуют и стандартные способы выделения памяти в языке Си (с помощью функций malloc и calloc), но они не очень удобны.

Следующая простейшая программа, которая использует динамический массив, вводит с клавиатуры размер массива, все его элементы, а затем сортирует их и выводит на экран.

динамические массивы используются тогда, когда на момент написания программы размер массива неизвестен

для того, чтобы работать с динамическим массивом, надо объявить указатель соответствующего типа (в нем будет храниться адрес первого элемента массива);

**1.5 Стандартные функции работы с файлами.**

Файл – это именованная область ячеек памяти, в которой хранятся данные одного типа. Файл имеет следующие характерные особенности

Файлы бывают текстовыми и двоичными. Текстовый файл – файл, в котором каждый символ из используемого набора хранится в виде одного байта (код, соответствующий символу). Текстовые файлы разбиваются на несколько строк с помощью специального символа "конец строки".

Текстовый файл заканчивается специальным символом "конец файла". Двоичный файл – файл, данные которого представлены в бинарном виде. При записи в двоичный файл символы и числа записываются в виде последовательности байт (в своем внутреннем двоичном представлении в памяти компьютера).

**1.5.1 Функция открытия файла**

При открытии файла (потока) в программу возвращается указатель на поток (файловый указатель), являющийся указателем на объект структурного типа FILE. Этот указатель идентифицирует поток во всех последующих операциях.

**1.5.2 Функция закрытия файла**

Открытые на диске файлы после окончания работы с ними рекомендуется закрыть явно. Это является хорошим тоном в программировании.

**1.5.3 Функция контроля конца файла**

Для контроля достижения конца файла есть функция feof. int feof(FILE \* filename);

**1.6 Динамические структуры данных**

Часто в серьезных программах надо использовать данные, размер и структура которых должны меняться в процессе работы. Динамические массивы здесь не выручают, поскольку заранее нельзя сказать, сколько памяти надо выделить – это выясняется только в процессе работы. Например, надо проанализировать текст и определить, какие слова и в каком количество в нем встречаются, причем эти слова нужно расставить по алфавиту. В таких случаях применяют данные особой структуры, которые представляют собой отдельные элементы, связанные с помощью ссылок.

**1.7 Линейный список**

В простейшем случае каждый узел содержит всего одну ссылку. Для определенности будем считать, что решается задача частотного анализа текста – определения всех слов, встречающихся в тексте и их количества. В этом случае область данных элемента включает строку (длиной не более 40 символов) и целое число.

**1.8 Парадигмы ООП – инкапсуляция, наследование, полиморфизм**

Инкапсуляция - сведение кода и данных воедино в одном объекте, получившим название класс. Наследование - наличие в языке ООП механизма, позволяющего объектам класса наследовать характеристики более простых и общих типов. Наследование обеспечивает как требуемый уровень общности, так и необходимую специализацию.

Полиморфизм - дословный перевод с греческого "много форм". В С++ полиморфизм реализуется с помощью виртуальных функций, которые позволяют в рамках всей иерархии классов иметь несколько версий одной и той же функции. Решение о том, какая именно версия должна выполняться в данный момент, определяется на этапе выполнения программы и носит название позднего связывания

**1.9 Использование динамических массивов.**

Массивы, создаваемые в динамической памяти, будем называть динамическими (размерность становится известна в процессе выполнения программы). При описании массива после имени в квадратных скобках задается количество его элементов (размерность), например int a[10]. Размерность массива может быть задана только константой или константным выражением.

Обращение к элементу динамического массива осуществляется так же, как и к элементу обычного. Если динамический массив в какой-то момент работы программы перестает быть нужным и мы собираемся впоследствии использовать эту память повторно, необходимо освободить ее с помощью операции delete[], например: delete [] a; (размерность массива при этом не указывается)

**1.10 Классы. Программирование линейных алгоритмов с использованием функций инициализации set()**

Для каждого объекта класса устанавливается область видимости либо явно – указанием уровня доступа одним из ключевых слов public, private, protected с двоеточием, либо неявно – по умолчанию. Указание области видимости относится ко всем последующим объектам класса, пока не встретится указание другой области видимости. Область видимости public разрешает доступ к объектам класса из любой части программы, в которой известен этот объект (общедоступный). Область видимости private разрешает доступ к объектам класса только из методов этого класса.

**1.11 Конструктор класса**

Конструктор – это метод класса, имя которого совпадает с именем класса. Конструктор вызывается автоматически после выделения памяти для переменной и обеспечивает инициализацию компонент – данных.

**1.12 Деструктор класса**

Еще одним специальным методом класса является деструктор. Деструктор вызывается перед освобождением памяти, занимаемой объектной переменной, и предназначен для выполнения дополнительных действий, связанных с уничтожением объектной переменной, например, для освобождения динамической памяти, закрытия, уничтожения файлов и т.п.

Дружественная функция (friend)

**1.13 Перегрузка операций**

Наличие в классе конструктора String:: String(String&) и операторов присваивания позволяет защитить объекты класса от побитового копирования.

**1.14 Наследование классов**

Наследование - механизм создания производного класса из базового. Т.е., к существующему классу можно что-либо добавить, или изменять его каким-либо образом для создания нового (производного) класса. Это мощный механизм для повторного использования кода. Наследование позволяет создавать иерархию связанных типов, совместно использующих код и интерфейс. Модификатор прав доступа используется для изменения доступа к наследуемым объектам в соответствии с правилами, указанными в таблице

# 2 ПРАКТИЧЕСКАЯ ЧАСТЬ

## 2.1 Структура программы



Рисунок 2 – Диаграмма классов

## 2.2 Алгоритм решения задачи

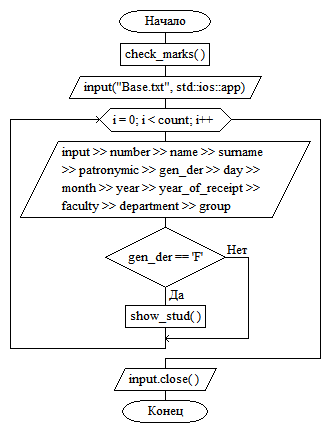


Рисунок 3 – Меню выбора пола

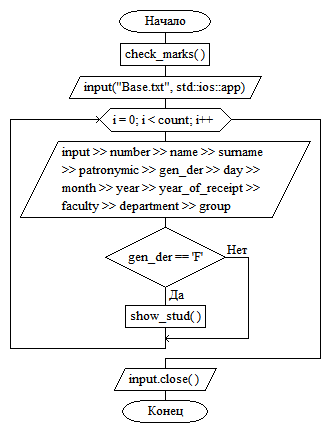


Рисунок 4 – Условие на проверку женского пола

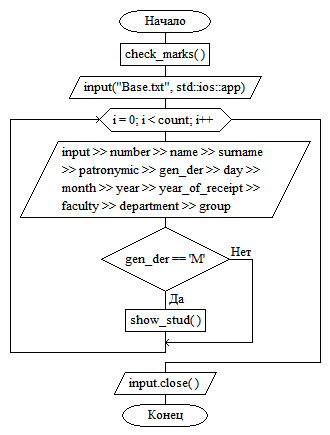


Рисунок 5 – Условие на проверку мужского пола

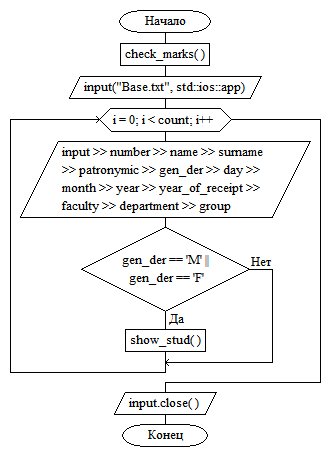


Рисунок 6 – Условие вывода мужского и женского пола

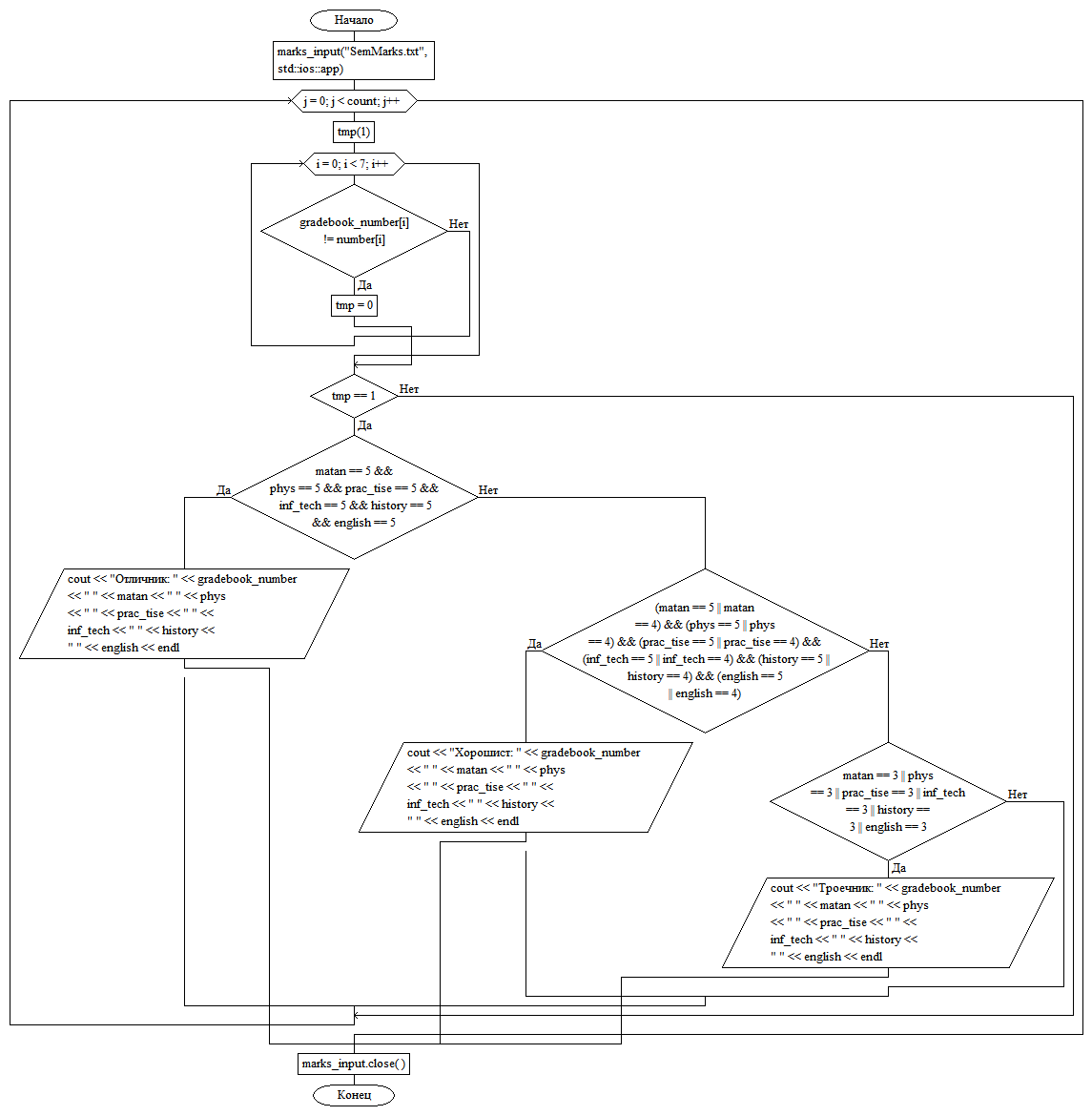


Рисунок 7 – Условие проверки учебного показателя (“Хорошист, Отличник, Троечник”)

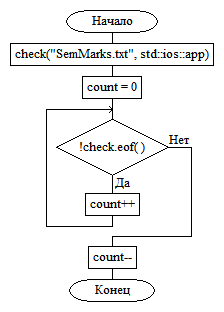


Рисунок 8 – Условие проверки количества оценок

На рисунке 3 представлено меню выбора пола при вводе параметров 1,2 или 3 мы попадаем на метки мужского или женского пола, женского или мужского соответственно. На рисунках 4,5,6 блок-схемы с проверкой пола в бд студентов, на рисунке 7 условие проверки на показатель “Хорошист, Отличник, Троечник”, программа пробегается по нашей бд студентов, на Рисунке 8 счётчик количества оценок, и далее фильтрует по критериям “Хорошист, Отличник или Троечник”

# Анализ результатов

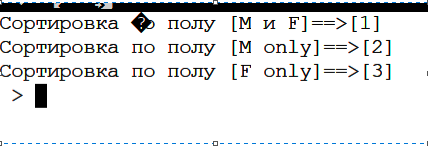


Рисунок 9 – Меню выбора пола

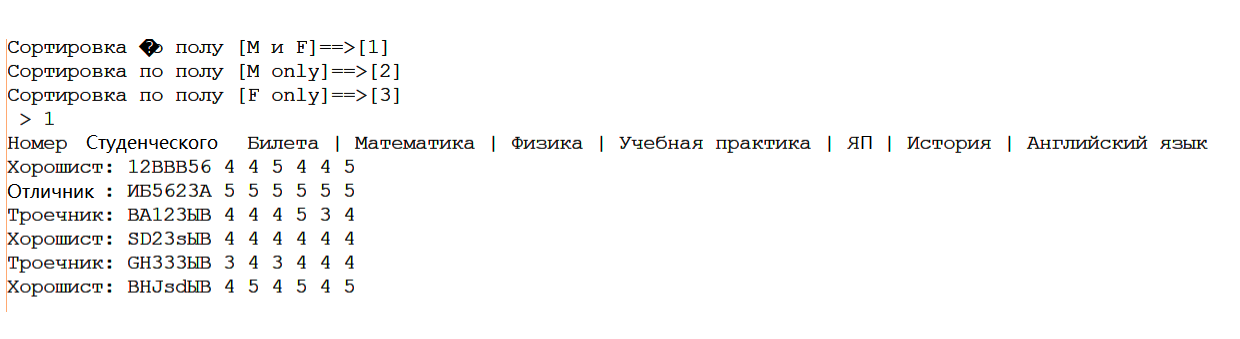


Рисунок 10 – Вывод студентов Мужского и Женского пола

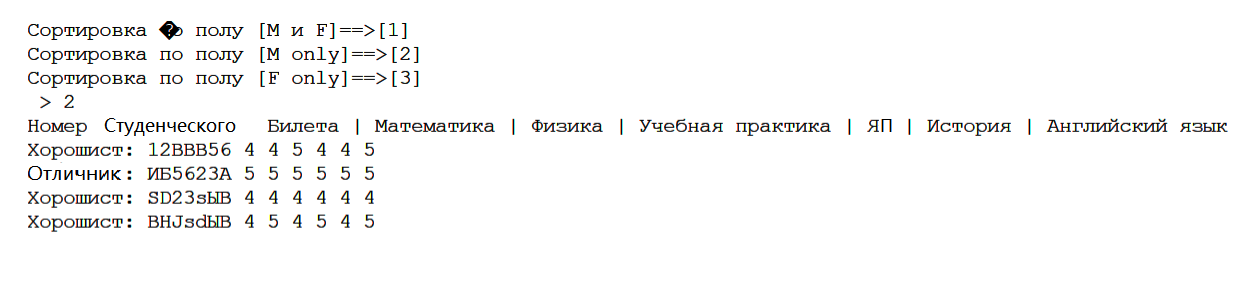


Рисунок 11 – Вывод студентов Мужского пола

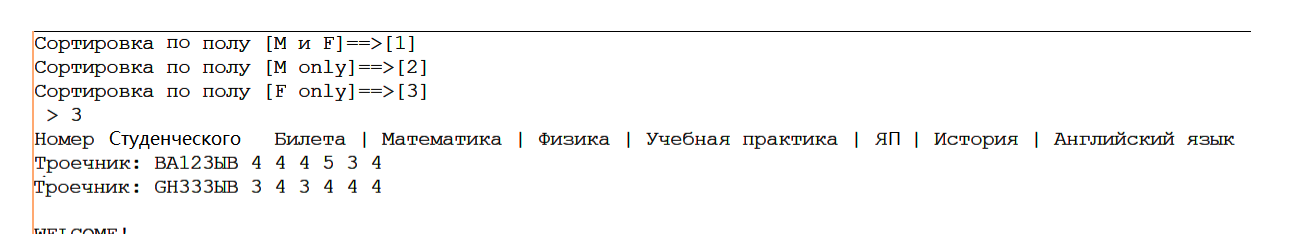


Рисунок 12 – Вывод студентов Женского пола

# Листинг кода решения задачи



Рисунок 13 – Скриншот с кодом на выполнение варианта 46 часть 1

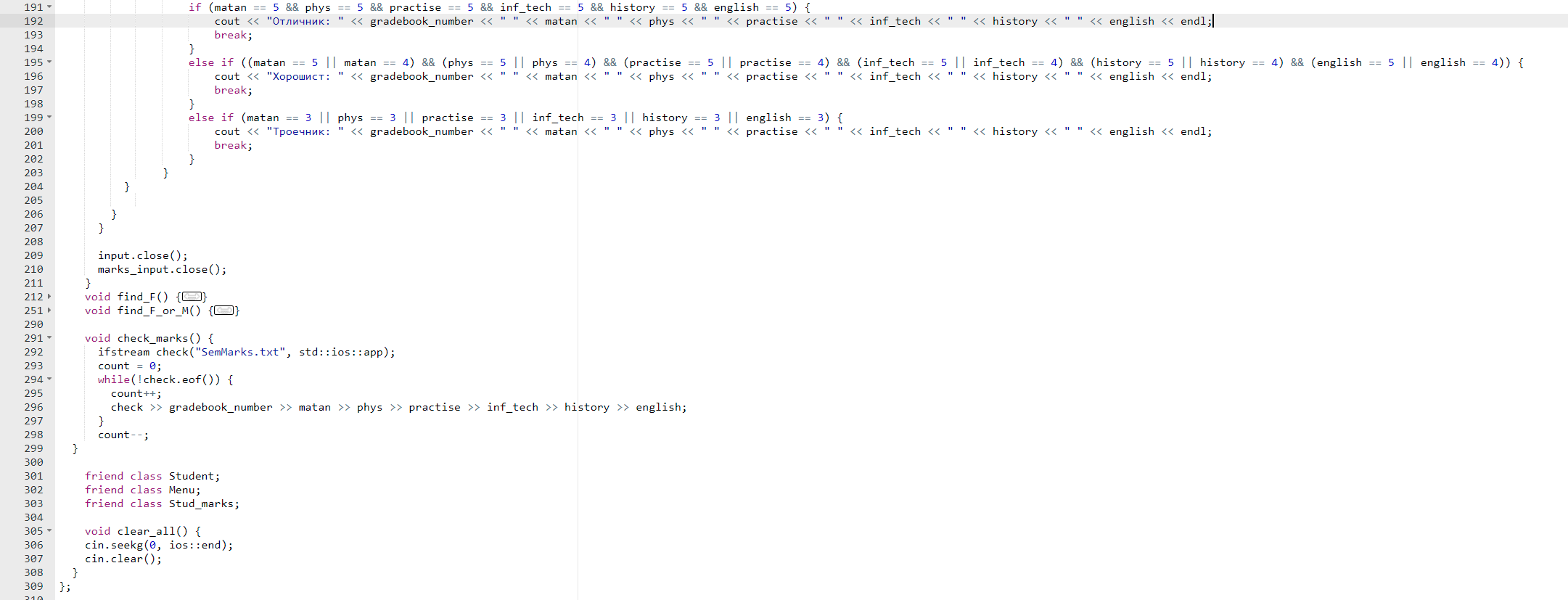


Рисунок 14 – Скриншот с кодом на выполнение варианта 46 часть 2

# ЗАКЛЮЧЕНИЕ

При выполнении курсовой работы, были закреплены навыки, полученные при изучении дисциплины Языки программирования и языка программирования C++. Было выполнено общее здание по проектированию структуры программы, а также внедрение в ООП. В итоге курсовой работы имеется программа, соответствующую требованиям проектирования, а также работа с вариантом, при вводе данных в который, получается верный ответ согласно варианту.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Мерсов А. А. Основы объектно-ориентированного программирования на языке С++ [Электронный ресурс]: практикум / А. А. Мерсов, А. М. Русаков, В. В. Филатов. — М.: РТУ МИРЭА, 2021. — Электрон. опт. диск (ISO)
2. Павловская Т. А. C/C++. Программирование на языке высокого уровня: учебник для вузов. – " Издательский дом"" Питер""", 2021.
3. Страуструп Б. Программирование. Принципы и практика использования C++. – Litres, 2021.
4. Лафоре Р. Объектно-ориентированное программирование в C++. 4-е изд., перераб. и доп //Санкт-Петербург.: Питер. – 2018.
5. Стенли Липпман Язык программирования С++: полное руководство / Липпман Стенли, ЛажойеЖози. – Саратов: Профобразование, 2017. –1104 c.
6. Страуструп Б. Дизайн и эволюция языка С++. – Litres, 2022.

# ПРИЛОЖЕНИЕ

Листинг программы:

#include <stdio.h>

#include <iostream>

#include <fstream>

#include <iomanip>

using namespace std**;**

**class** **Core** **{**

public**:**

**int** protect\_marks**()** **{**

**int** val**;**

**while** **(**true**)** **{**

cin **>>** val**;**

**if** **(**val **>** 0 **&&** val **<=** 5**)** **{**

clear\_all**();**

**return** val**;**

**}**

cout **<<** "[CODE\_ERROR] - Введены неверные данные" **<<** endl**;**

clear\_all**();**

**}**

**}**

char protect**()** **{**

char val**;**

**int** result**;**

**while** **(**true**)** **{**

cin **>>** val**;**

result **=** val **-** '0'**;**

**if** **(!**isdigit**(**val**))** **{**

cout **<<** "[CODE\_ERROR] - Введены неверные данные" **<<** endl**;**

**return** 8**;**

**}**

**else** **if** **(**result **>** 0 **&&** result **<=** 9**)** **{**

clear\_all**();**

**return** result**;**

**}**

cout **<<** "[CODE\_ERROR] - Введены неверные данные" **<<** endl**;**

clear\_all**();**

**}**

**}**

**int** protect\_change**()** **{**

**int** val**;**

**while** **(**true**)** **{**

cin **>>** val**;**

**if** **(**val **>** 0 **&&** val **<=** 4**)** **{**

clear\_all**();**

**return** val**;**

**}**

cout **<<** "[CODE\_ERROR] - Введены неверные данные" **<<** endl**;**

clear\_all**();**

**}**

**}**

**int** protect\_task**()** **{**

**int** val**;**

**while** **(**true**)** **{**

cin **>>** val**;**

**if** **(**val **>** 0 **&&** val **<=** 5**)** **{**

clear\_all**();**

**return** val**;**

**}**

cout **<<** "[CODE\_ERROR] - Введены неверные данные" **<<** endl**;**

clear\_all**();**

**}**

**}**

**int** protect\_day**()** **{**

**int** val**;**

**while** **(**true**)** **{**

cin **>>** val**;**

**if** **(**val **>** 0 **&&** val **<=** 31**)** **{**

clear\_all**();**

**return(**val**);**

**}**

cout **<<** "[CODE\_ERROR] - Введены неверные данные" **<<** endl**;**

cout **<<** "Неверная Дата" **<<** endl**;**

clear\_all**();**

**}**

**}**

**int** protect\_month**()** **{**

**int** val**;**

**while** **(**true**)** **{**

cin **>>** val**;**

**if** **(**val **>** 0 **&&** val **<=** 12**)** **{**

clear\_all**();**

**return(**val**);**

**}**

cout **<<** "[CODE\_ERROR] - Введены неверные данные" **<<** endl**;**

cout **<<** "Неверная Дата" **<<** endl**;**

clear\_all**();**

**}**

**}**

**int** protect\_year**()** **{**

**int** val**;**

**while** **(**true**)** **{**

cin **>>** val**;**

**if** **(**val **>** 1922 **&&** val **<=** 2022**)** **{**

clear\_all**();**

**return(**val**);**

**}**

cout **<<** "[CODE\_ERROR] - Введены неверные данные" **<<** endl**;**

cout **<<** "Неверная Дата" **<<** endl**;**

clear\_all**();**

**}**

**}**

char protect\_ch**()** **{**

char val**;**

**while** **(**true**)** **{**

cin **>>** val**;**

**if** **(**val **==** 'M' **||** val **==** 'F'**)** **{**

clear\_all**();**

**return** val**;**

**}**

cout **<<** "[CODE\_ERROR] - Введены неверные данные" **<<** endl**;**

clear\_all**();**

**}**

**}**

void clear\_all**()** **{**

cin**.**seekg**(**0**,** ios**::**end**);**

cin**.**clear**();**

**}**

**};**

**class** **Task** **:** Core **{**

private**:**

**int** command**;**

**int** a **=** 0**;**

**int** b **=** 0**;**

**int** c **=** 0**;**

char gradebook\_number**[**32**];**

char number**[**32**];**

**int** matan**;**

**int** phys**;**

**int** prac\_tise**;**

**int** inf\_tech**;**

**int** history**;**

**int** english**;**

**int** count **=** 0**;**

char surname**[**32**];**

char name**[**32**];**

char patronymic**[**32**];**

**int** day**;**

**int** month**;**

**int** year**;**

**int** year\_of\_receipt**;**

char gen\_der**;**

char faculty**[**32**];**

char department**[**32**];**

char group**[**32**];**

public**:**

void task\_menu**()** **{**

check\_marks**();**

cout **<<** "Сортировка по полу [M и F]==>[1]" **<<** endl**;**

cout **<<** "Сортировка по полу [M only]==>[2]" **<<** endl**;**

cout **<<** "Сортировка по полу [F only]==>[3]" **<<** endl**;**

cout **<<** " > "**;**

cin **>>** command**;** clear\_all**();**

cout **<<** "Номер Студенческого Билета | Математика | Физика | Учебная практика | ЯП | История | Английский язык" **<<** endl**;**

switch **(**command**)** **{**

case 1**:**

**{**

find\_F\_or\_M**();**

**break;**

**}**

case 2**:**

**{**

find\_M**();**

**break;**

**}**

case 3**:**

**{**

find\_F**();**

**break;**

**}**

**}**

**}**

void show\_stud**()** **{**

ifstream marks\_input**(**"SemMarks.txt"**,** std**::**ios**::**app**);**

**for** **(int** j **=** 0**;** j **<** count**;** j**++)** **{**

marks\_input **>>** gradebook\_number **>>** matan **>>** phys **>>** prac\_tise **>>** inf\_tech **>>** history **>>** english**;**

**int** tmp**(**1**);**

**for** **(int** i **=** 0**;** i **<** 7**;** i**++)** **{**

**if** **(**gradebook\_number**[**i**]** **!=** number**[**i**])** **{**

tmp **=** 0**;**

**break;**

**}**

**}**

**if** **(**tmp **==** 1**)** **{**

**if** **(**matan **==** 5 **&&** phys **==** 5 **&&** prac\_tise **==** 5 **&&** inf\_tech **==** 5 **&&** history **==** 5 **&&** english **==** 5**)** **{**

cout **<<** "Отличник: " **<<** gradebook\_number **<<** " " **<<** matan **<<** " " **<<** phys **<<** " " **<<** prac\_tise **<<** " " **<<** inf\_tech **<<** " " **<<** history **<<** " " **<<** english **<<** endl**;**

**break;**

**}**

**else** **if** **((**matan **==** 5 **||** matan **==** 4**)** **&&** **(**phys **==** 5 **||** phys **==** 4**)** **&&** **(**prac\_tise **==** 5 **||** prac\_tise **==** 4**)** **&&** **(**inf\_tech **==** 5 **||** inf\_tech **==** 4**)** **&&** **(**history **==** 5 **||** history **==** 4**)** **&&** **(**english **==** 5 **||** english **==** 4**))** **{**

cout **<<** "Хорошист: " **<<** gradebook\_number **<<** " " **<<** matan **<<** " " **<<** phys **<<** " " **<<** prac\_tise **<<** " " **<<** inf\_tech **<<** " " **<<** history **<<** " " **<<** english **<<** endl**;**

**break;**

**}**

**else** **if** **(**matan **==** 3 **||** phys **==** 3 **||** prac\_tise **==** 3 **||** inf\_tech **==** 3 **||** history **==** 3 **||** english **==** 3**)** **{**

cout **<<** "Троечник: " **<<** gradebook\_number **<<** " " **<<** matan **<<** " " **<<** phys **<<** " " **<<** prac\_tise **<<** " " **<<** inf\_tech **<<** " " **<<** history **<<** " " **<<** english **<<** endl**;**

**break;**

**}**

**}**

**}**

marks\_input**.**close**();**

**}**

void find\_M**()** **{**

check\_marks**();**

ifstream **input(**"Base.txt"**,** std**::**ios**::**app**);**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

**input** **>>** number **>>** name **>>** surname **>>** patronymic **>>** gen\_der **>>** day **>>** month **>>** year **>>** year\_of\_receipt **>>** faculty **>>** department **>>** group**;**

**if** **(**gen\_der **==** 'M'**)** **{**

show\_stud**();**

**}**

**}**

**input.**close**();**

**}**

void find\_F**()** **{**

check\_marks**();**

ifstream **input(**"Base.txt"**,** std**::**ios**::**app**);**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

**input** **>>** number **>>** name **>>** surname **>>** patronymic **>>** gen\_der **>>** day **>>** month **>>** year **>>** year\_of\_receipt **>>** faculty **>>** department **>>** group**;**

**if** **(**gen\_der **==** 'F'**)** **{**

show\_stud**();**

**}**

**}**

**input.**close**();**

**}**

void find\_F\_or\_M**()** **{**

check\_marks**();**

ifstream **input(**"Base.txt"**,** std**::**ios**::**app**);**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

**input** **>>** number **>>** name **>>** surname **>>** patronymic **>>** gen\_der **>>** day **>>** month **>>** year **>>** year\_of\_receipt **>>** faculty **>>** department **>>** group**;**

**if** **(**gen\_der **==** 'M' **||** gen\_der **==** 'F'**)** **{**

show\_stud**();**

**}**

**}**

**input.**close**();**

**}**

void check\_marks**()** **{**

ifstream check**(**"SemMarks.txt"**,** std**::**ios**::**app**);**

count **=** 0**;**

**while** **(!**check**.**eof**())** **{**

count**++;**

check **>>** gradebook\_number **>>** matan **>>** phys **>>** prac\_tise **>>** inf\_tech **>>** history **>>** english**;**

**}**

count**--;**

**}**

friend **class** **Student;**

friend **class** **Menu;**

friend **class** **Stud\_marks;**

void clear\_all**()** **{**

cin**.**seekg**(**0**,** ios**::**end**);**

cin**.**clear**();**

**}**

**};**

**class** **Student** **:** Core **{**

private**:**

char**\*** surname **=** nullptr**;**

char**\*** name **=** nullptr**;**

char**\*** patronymic **=** nullptr**;**

**int** day**;**

**int** month**;**

**int** year**;**

char**\*** gen\_der **=** nullptr**;**

**int** year\_of\_receipt**;**

char**\*** faculty **=** nullptr**;**

char**\*** department **=** nullptr**;**

char**\*** group **=** nullptr**;**

char**\*** gradebook\_number **=** nullptr**;**

public**:**

Student**()** **{**

surname **=** new char**[**31**]();**

name **=** new char**[**31**]();**

patronymic **=** new char**[**31**]();**

gen\_der **=** new char**(**'\0'**);**

day **=** 0**;**

month **=** 0**;**

year **=** 0**;**

year\_of\_receipt **=** 0**;**

faculty **=** new char**[**32**];**

department **=** new char**[**32**];**

group **=** new char**[**32**];**

gradebook\_number **=** new char**[**32**];**

**}**

Student**(**

char**\*** surname**,**

char**\*** name**,**

char**\*** patronymic**,**

**int** day**,**

**int** month**,**

**int** year**,**

**int** year\_of\_receipt**,**

char**\*** gen\_der**,**

char**\*** faculty**,**

char**\*** department**,**

char**\*** group**,**

char**\*** gradebook\_number**)**

**{**

surname **=** new char**[**31**]();**

name **=** new char**[**31**]();**

patronymic **=** new char**[**31**]();**

gen\_der **=** new char**(**'\0'**);**

day **=** 0**;**

month **=** 0**;**

year **=** 0**;**

year\_of\_receipt **=** 0**;**

faculty **=** new char**[**32**];**

department **=** new char**[**32**];**

group **=** new char**[**32**];**

gradebook\_number **=** new char**[**32**];**

this**->**surname **=** surname**;**

this**->**name **=** name**;**

this**->**patronymic **=** patronymic**;**

this**->**day **=** day**;**

this**->**month **=** month**;**

this**->**year **=** year**;**

this**->**year\_of\_receipt **=** year\_of\_receipt**;**

this**->**faculty **=** faculty**;**

this**->**department **=** department**;**

this**->**group **=** group**;**

this**->**gradebook\_number **=** gradebook\_number**;**

this**->**gen\_der **=** gen\_der**;**

**}**

void **set()** **{**

set\_name**();**

set\_surname**();**

set\_patronymic**();**

set\_gen\_der**();**

set\_birthdate**();**

set\_year\_of\_receipt**();**

set\_faculty**();**

set\_department**();**

set\_group**();**

set\_gradebook\_number**();**

**}**

void write**()** **{**

ofstream output**(**"Base.txt"**,** std**::**ios**::**app**);**

output **<<** gradebook\_number **<<** " " **<<** name **<<** " " **<<** surname **<<** " " **<<** patronymic **<<** " " **<<** gen\_der **<<** " " **<<** day **<<** " " **<<** month **<<** " " **<<** year **<<** " " **<<** year\_of\_receipt **<<** " " **<<** faculty **<<** " " **<<** department **<<** " " **<<** group **<<** endl**;**

output**.**close**();**

**}**

void set\_name**()** **{**

cout **<<** "Введите Имя Студента: "**;**

cin **>>** name**;**

clear\_all**();**

**}**

void set\_surname**()** **{**

cout **<<** "Введите Фамилию Студента: "**;**

cin **>>** surname**;**

clear\_all**();**

**}**

void set\_patronymic**()** **{**

cout **<<** "Введите Отчество Студента: "**;**

cin **>>** patronymic**;**

clear\_all**();**

**}**

void set\_gen\_der**()** **{**

cout **<<** "Введите Пол Студента: "**;**

**\***gen\_der **=** protect\_ch**();**

**}**

void set\_birthdate**()** **{**

cout **<<** "Введите Дату Рождения Студента- [День Месяц год]: "**;**

day **=** protect\_day**();**

month **=** protect\_month**();**

year **=** protect\_year**();**

**}**

void set\_year\_of\_receipt**()** **{**

cout **<<** "Введите Дату Поступления Студента "**;**

year\_of\_receipt **=** protect\_year**();**

**}**

void set\_faculty**()** **{**

cout **<<** "Введите Название Факультета "**;**

cin **>>** faculty**;**

clear\_all**();**

**}**

void set\_department**()** **{**

cout **<<** "Введите Название Кафедры "**;**

cin **>>** department**;**

clear\_all**();**

**}**

void set\_group**()** **{**

cout **<<** "Введите Группу Студента "**;**

cin **>>** group**;**

clear\_all**();**

**}**

void set\_gradebook\_number**()** **{**

cout **<<** "Введите Номер Студенческого Билета "**;**

cin **>>** gradebook\_number**;**

clear\_all**();**

**}**

void get**()** **{**

get\_name**();**

get\_surname**();**

get\_patronymic**();**

get\_gen\_der**();**

get\_birthdate**();**

get\_year\_of\_receipt**();**

get\_faculty**();**

get\_department**();**

get\_group**();**

get\_gradebook\_number**();**

**}**

void get\_name**()** **{**

cout **<<** endl **<<** "Имя Студента: "**;**

cout **<<** name**;**

**}**

void get\_surname**()** **{**

cout **<<** endl **<<** "Фамилия Студента: "**;**

cout **<<** surname**;**

**}**

void get\_patronymic**()** **{**

cout **<<** endl **<<** "Отчество Студента: "**;**

cout **<<** patronymic**;**

**}**

void get\_gen\_der**()** **{**

cout **<<** endl **<<** "Пол Студента: "**;**

cout **<<** gen\_der**;**

**}**

void get\_birthdate**()** **{**

cout **<<** endl **<<** "Дата Рождения Студента -[Д М Г]: "**;**

cout **<<** day **<<** " " **<<** month **<<** " " **<<** year**;**

**}**

void get\_year\_of\_receipt**()** **{**

cout **<<** endl **<<** "Дата Поступления Студента: "**;**

cout **<<** year\_of\_receipt**;**

**}**

void get\_faculty**()** **{**

cout **<<** endl **<<** "Факультет Студента: "**;**

cout **<<** faculty**;**

**}**

void get\_department**()** **{**

cout **<<** endl **<<** "Кафедра Студента: "**;**

cout **<<** department**;**

**}**

void get\_group**()** **{**

cout **<<** endl **<<** "Группа Студента: "**;**

cout **<<** group**;**

**}**

void get\_gradebook\_number**()** **{**

cout **<<** endl **<<** "Номер Студенческого Билета: "**;**

cout **<<** gradebook\_number**;**

**}**

char ret\_gradebook\_number**()** **{**

**return** **\***gradebook\_number**;**

**}**

char ret\_name**()** const **{**

**return** **\***name**;**

**}**

char ret\_surname**()** const **{**

**return** **\***surname**;**

**}**

char ret\_patronymic**()** const **{**

**return** **\***patronymic**;**

**}**

char ret\_gen\_der**()** const **{**

**return** **\***gen\_der**;**

**}**

**int** ret\_day**()** const **{**

**return** day**;**

**}**

**int** ret\_month**()** const **{**

**return** month**;**

**}**

**int** ret\_year**()** const **{**

**return** year**;**

**}**

**int** ret\_year\_of\_receipt**()** const **{**

**return** year\_of\_receipt**;**

**}**

char ret\_faculty**()** const **{**

**return** **\***faculty**;**

**}**

char ret\_department**()** const **{**

**return** **\***department**;**

**}**

char ret\_group**()** const **{**

**return** **\***group**;**

**}**

char ret\_gradebook\_number**()** const **{**

**return** **\***gradebook\_number**;**

**}**

void clear\_all**()** **{**

cin**.**seekg**(**0**,** ios**::**end**);**

cin**.**clear**();**

**}**

**~**Student**()** **{**

delete surname**;**

delete name**;**

delete patronymic**;**

delete faculty**;**

delete department**;**

delete group**;**

delete gradebook\_number**;**

**}**

**};**

**class** **Stud\_marks** **:** Core **{**

private**:**

**int** count**;**

char gradebook\_number**[**32**];**

char number**[**32**];**

**int** command**;**

**int** matan**;**

**int** phys**;**

**int** prac\_tise**;**

**int** inf\_tech**;**

**int** history**;**

**int** english**;**

public**:**

friend **class** **Student;**

friend **class** **Core;**

friend **class** **Menu;**

Stud\_marks**()** **{**

count **=** 0**;**

check\_marks**();**

**}**

void add\_mark**()** **{**

count**++;**

cout **<<** "Введите Номер Студенческого Билета: " **<<** endl**;**

cin **>>** gradebook\_number**;** clear\_all**();**

cout **<<** "Введите Оценку По Математике: "**;**

cin **>>** matan**;** clear\_all**();**

cout **<<** "Введите Оценку По Физике: "**;**

cin **>>** phys**;** clear\_all**();**

cout **<<** "Введите Оценку По Учебной практике: "**;**

cin **>>** prac\_tise**;** clear\_all**();**

cout **<<** "Введите Оценку По ЯП: "**;**

cin **>>** inf\_tech**;** clear\_all**();**

cout **<<** "Введите Оценку По История: "**;**

cin **>>** history**;** clear\_all**();**

cout **<<** "Введите Оценку По Английский Язык: "**;**

cin **>>** english**;** clear\_all**();**

ofstream output**(**"SemMarks.txt"**,** std**::**ios**::**app**);**

output **<<** gradebook\_number **<<** " " **<<** matan **<<** " " **<<** phys **<<** " " **<<** prac\_tise **<<** " " **<<** inf\_tech **<<** " " **<<** history **<<** " " **<<** english **<<** endl**;**

output**.**close**();**

**}**

void delete\_mark**()** **{**

ifstream **input(**"SemMarks.txt"**,** std**::**ios**::**app**);**

ofstream output**(**"Replaser.txt"**);**

cout **<<** endl **<<** "Введите Номер Студенческого Билета: " **<<** endl**;**

cin **>>** number**;**

clear\_all**();**

check\_marks**();**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

**input** **>>** gradebook\_number **>>** matan **>>** phys **>>** prac\_tise **>>** inf\_tech **>>** history **>>** english**;**

**int** tmp**(**1**);**

**for** **(int** i **=** 0**;** i **<** 7**;** i**++)** **{**

**if** **(**gradebook\_number**[**i**]** **!=** number**[**i**])** **{**

tmp **=** 0**;**

**break;**

**}**

**}**

**if** **(**tmp **==** 0**)** **{**

output **<<** gradebook\_number **<<** " " **<<** matan **<<** " " **<<** phys **<<** " " **<<** prac\_tise **<<** " " **<<** inf\_tech **<<** " " **<<** history **<<** " " **<<** english **<<** endl**;**

**}**

**}**

ifstream if\_rewrite**(**"Replaser.txt"**,** std**::**ios**::**app**);**

ofstream of\_rewrite**(**"SemMarks.txt"**);**

count **-=** 1**;**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

if\_rewrite **>>** gradebook\_number **>>** matan **>>** phys **>>** prac\_tise **>>** inf\_tech **>>** history **>>** english**;**

of\_rewrite **<<** gradebook\_number **<<** " " **<<** matan **<<** " " **<<** phys **<<** " " **<<** prac\_tise **<<** " " **<<** inf\_tech **<<** " " **<<** history **<<** " " **<<** english **<<** endl**;**

**}**

cout **<<** "Оценки Студента Удалены Из Списка" **<<** endl**;**

if\_rewrite**.**close**();**

of\_rewrite**.**close**();**

**input.**close**();**

output**.**close**();**

**}**

void Change\_Marks**()** **{**

ifstream **input(**"SemMarks.txt"**,** std**::**ios**::**app**);**

ofstream output**(**"Replaser.txt"**);**

cout **<<** endl **<<** "Введите Номер Студенческого Билета: " **<<** endl**;**

cin **>>** number**;**

clear\_all**();**

cout **<<** endl**;**

cout **<<** "Что Вы Хотите Изменить? " **<<** endl **<<** endl**;**

cout **<<** "[1] Номер Студенческого Билета" **<<** endl**;**

cout **<<** "[2] Математика" **<<** endl**;**

cout **<<** "[3] Физика" **<<** endl**;**

cout **<<** "[4] Учебная практика" **<<** endl**;**

cout **<<** "[5] ЯП" **<<** endl**;**

cout **<<** "[6] История" **<<** endl**;**

cout **<<** "[7] Английский Язык" **<<** endl**;**

cout **<<** "[8] Назад" **<<** endl**;**

cout **<<** " > "**;**

command **=** protect**();**

check\_marks**();**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

**input** **>>** gradebook\_number **>>** matan **>>** phys **>>** prac\_tise **>>** inf\_tech **>>** history **>>** english**;**

**int** tmp**(**1**);**

**for** **(int** i **=** 0**;** i **<** 7**;** i**++)** **{**

**if** **(**gradebook\_number**[**i**]** **!=** number**[**i**])** **{**

tmp **=** 0**;**

**break;**

**}**

**}**

**if** **(**tmp **==** 1**)** **{**

switch **(**command**)** **{**

case 1**:**

**{**

cout **<<** "Введите Новый Номер Студенческого Билета: "**;**

cin **>>** gradebook\_number**;**

clear\_all**();**

**break;**

**}**

case 2**:**

**{**

cout **<<** "Введите Оценку По Математике: "**;**

cin **>>** matan**;** clear\_all**();**

**break;**

**}**

case 3**:**

**{**

cout **<<** "Введите Оценку По Физике: "**;**

cin **>>** phys**;** clear\_all**();**

**break;**

**}**

case 4**:**

**{**

cout **<<** "Введите Оценку По Учебной практике: "**;**

cin **>>** prac\_tise**;** clear\_all**();**

**break;**

**}**

case 5**:**

**{**

cout **<<** "Введите Оценку По ЯП: "**;**

cin **>>** inf\_tech**;** clear\_all**();**

**break;**

**}**

case 6**:**

**{**

cout **<<** "Введите Оценку По История: "**;**

cin **>>** history**;** clear\_all**();**

**}**

case 7**:**

**{**

cout **<<** "Введите Оценку По Английский Язык: "**;**

cin **>>** english**;** clear\_all**();**

**}**

case 8**:**

**{**

**break;**

**}**

**}**

output **<<** gradebook\_number **<<** " " **<<** matan **<<** " " **<<** phys **<<** " " **<<** prac\_tise **<<** " " **<<** inf\_tech **<<** " " **<<** history **<<** " " **<<** english **<<** endl**;**

**}**

**else** output **<<** gradebook\_number **<<** " " **<<** matan **<<** " " **<<** phys **<<** " " **<<** prac\_tise **<<** " " **<<** inf\_tech **<<** " " **<<** history **<<** " " **<<** english **<<** endl**;**

**}**

ifstream if\_rewrite**(**"Replaser.txt"**,** std**::**ios**::**app**);**

ofstream of\_rewrite**(**"SemMarks.txt"**);**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

if\_rewrite **>>** gradebook\_number **>>** matan **>>** phys **>>** prac\_tise **>>** inf\_tech **>>** history **>>** english**;**

of\_rewrite **<<** gradebook\_number **<<** " " **<<** matan **<<** " " **<<** phys **<<** " " **<<** prac\_tise **<<** " " **<<** inf\_tech **<<** " " **<<** history **<<** " " **<<** english **<<** endl**;**

**}**

cout **<<** "Оценки Студента Изменены" **<<** endl**;**

if\_rewrite**.**close**();**

of\_rewrite**.**close**();**

**input.**close**();**

output**.**close**();**

**}**

void Show\_Marks**()** **{**

check\_marks**();**

ifstream **input(**"SemMarks.txt"**,** std**::**ios**::**app**);**

cout **<<** "Номер Студенческого Билета | Математика | Физика | Учебная практика | ЯП | История | Английский язык" **<<** endl**;**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

**input** **>>** gradebook\_number **>>** matan **>>** phys **>>** prac\_tise **>>** inf\_tech **>>** history **>>** english**;**

cout **<<** gradebook\_number **<<** " " **<<** matan **<<** " " **<<** phys **<<** " " **<<** prac\_tise **<<** " " **<<** inf\_tech **<<** " " **<<** history **<<** " " **<<** english **<<** endl**;**

**}**

**input.**close**();**

**}**

void check\_marks**()** **{**

ifstream check**(**"SemMarks.txt"**,** std**::**ios**::**app**);**

count **=** 0**;**

**while** **(!**check**.**eof**())** **{**

count**++;**

check **>>** gradebook\_number **>>** matan **>>** phys **>>** prac\_tise **>>** inf\_tech **>>** history **>>** english**;**

**}**

count**--;**

**}**

**~**Stud\_marks**()** **{**

**}**

void clear\_all**()** **{**

cin**.**seekg**(**0**,** ios**::**end**);**

cin**.**clear**();**

**}**

**};**

**class** **Menu** **:** Core **{**

private**:**

**int** command**;**

**int** count**;**

char surname**[**32**];**

char name**[**32**];**

char patronymic**[**32**];**

**int** day**;**

**int** month**;**

**int** year**;**

**int** year\_of\_receipt**;**

char gen\_der**;**

char faculty**[**32**];**

char department**[**32**];**

char group**[**32**];**

char gradebook\_number**[**32**];**

char number**[**32**];**

public**:**

friend **class** **Core;**

friend **class** **Student;**

friend **class** **Task;**

friend **class** **Stud\_marks;**

Menu**()** **{**

count **=** 0**;**

**}**

**int** welcome**()** **{**

ifstream check**(**"Base.txt"**);**

count **=** 0**;**

**while** **(!**check**.**eof**())** **{**

count**++;**

check **>>** gradebook\_number **>>** name **>>** surname **>>** patronymic **>>** gen\_der **>>** day **>>** month **>>** year **>>** year\_of\_receipt **>>** faculty **>>** department **>>** group**;**

**}**

count**--;**

cout **<<** "WELCOME!" **<<** endl **<<** endl**;**

cout **<<** "[1] - Добавить студента" **<<** endl**;**

cout **<<** "[2] - Удалить студента" **<<** endl**;**

cout **<<** "[3] - Изменить данные студента" **<<** endl**;**

cout **<<** "[4] - Вывести всю базу студентов" **<<** endl**;**

cout **<<** "[5] - Найти студента по зачетке" **<<** endl**;**

cout **<<** "[6] - Выполнение задачи" **<<** endl**;**

cout **<<** "[7] - Редактор Оценок" **<<** endl**;**

cout **<<** "[8] - Очистить экран" **<<** endl**;**

cout **<<** "[9] - Выйти из программы" **<<** endl**;**

cout **<<** " > "**;**

**//**char check**[**32**];** cin **>>** check**;**

command **=** protect**();**

**if** **(**command **==** 1**)** **{**

Student student**;**

student**.set();**

student**.**write**();**

count**++;**

cout **<<** "Студент Добавлен в список" **<<** endl **<<** endl**;**

welcome**();**

**}**

**else** **if** **(**command **==** 2**)** **{**

Delete\_Student**();**

cout **<<** endl**;**

welcome**();**

**}**

**else** **if** **(**command **==** 3**)** **{**

Change\_Student**();**

cout **<<** endl**;**

welcome**();**

**}**

**else** **if** **(**command **==** 4**)** **{**

Show\_Students**();**

cout **<<** endl**;**

welcome**();**

**}**

**else** **if** **(**command **==** 5**)** **{**

Find\_Student**();**

cout **<<** endl**;**

welcome**();**

**}**

**else** **if** **(**command **==** 6**)** **{**

cout **<<** "\033[2J\033[1;1H"**;**

Task ttask**;**

ttask**.**task\_menu**();**

cout **<<** endl**;**

welcome**();**

**}**

**else** **if** **(**command **==** 7**)** **{**

cout **<<** "\033[2J\033[1;1H"**;**

Marks\_menu**();**

**}**

**else** **if** **(**command **==** 8**)** **{**

cout **<<** "\033[2J\033[1;1H"**;**

welcome**();**

**}**

**else** **if** **(**command **==** 9**)** **{**

**return** 0**;**

**}**

**return** 0**;**

**}**

void Marks\_menu**()** **{**

cout **<<** "[1] - Добавить Оценки Студента (По Студенческому Билету)" **<<** endl**;**

cout **<<** "[2] - Удалить Оценки Студента (По Студенческому Билету)" **<<** endl**;**

cout **<<** "[3] - Изменить Оценки Студента (По Студенческому Билету)" **<<** endl**;**

cout **<<** "[4] - Вывести весь список оценок" **<<** endl**;**

cout **<<** "[5] - Назад" **<<** endl**;**

cout **<<** " > "**;**

**int** command **=** protect\_marks**();**

switch **(**command**)** **{**

case 1**:** **{**

Stud\_marks Marks**;**

Marks**.**add\_mark**();**

cout **<<** endl**;**

Marks\_menu**();**

**break;**

**}**

case 2**:** **{**

Stud\_marks Marks**;**

Marks**.**delete\_mark**();**

cout **<<** endl**;**

Marks\_menu**();**

**break;**

**}**

case 3**:** **{**

cout **<<** endl**;**

Stud\_marks Marks**;**

Marks**.**Change\_Marks**();**

Marks\_menu**();**

**break;**

**}**

case 4**:** **{**

Stud\_marks Marks**;**

Marks**.**Show\_Marks**();**

cout **<<** endl**;**

Marks\_menu**();**

**break;**

**}**

case 5**:** **{**

cout **<<** "\033[2J\033[1;1H"**;**

welcome**();**

**break;**

**}**

**}**

**}**

void Show\_Students**()** **{**

ifstream **input(**"Base.txt"**);**

cout **<<** endl**;**

**int** counter **=** 0**;**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

**input** **>>** gradebook\_number **>>** name **>>** surname **>>** patronymic **>>** gen\_der **>>** day **>>** month **>>** year **>>** year\_of\_receipt **>>** faculty **>>** department **>>** group**;**

counter**++;**

cout **<<** "(" **<<** counter **<<** ")" **<<** endl**;**

cout **<<** "ФИО(пол): " **<<** surname **<<** " " **<<** name **<<** " " **<<** patronymic **<<** " (" **<<** gen\_der **<<** ") " **<<** endl**;**

cout **<<** "Дата Рождения, Дата поступления: " **<<** day **<<** " " **<<** month **<<** " " **<<** year **<<** ", " **<<** year\_of\_receipt **<<** endl**;**

cout **<<** "Номер Студенческого Билета, Группа, Факультет, Кафедра: " **<<** gradebook\_number **<<** " " **<<** group **<<** " " **<<** faculty **<<** " " **<<** department **<<** endl **<<** endl**;**

**}**

**input.**close**();**

**}**

void Delete\_Student**()** **{**

ifstream **input(**"Base.txt"**,** std**::**ios**::**app**);**

ofstream output**(**"Replaser.txt"**);**

cout **<<** endl **<<** "Введите Номер Студенческого Билета: " **<<** endl**;**

cin **>>** number**;**

clear\_all**();**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

**input** **>>** gradebook\_number **>>** name **>>** surname **>>** patronymic **>>** gen\_der **>>** day **>>** month **>>** year **>>** year\_of\_receipt **>>** faculty **>>** department **>>** group**;**

**int** tmp**(**1**);**

**for** **(int** i **=** 0**;** i **<** 7**;** i**++)** **{**

**if** **(**gradebook\_number**[**i**]** **!=** number**[**i**])** **{**

tmp **=** 0**;**

**break;**

**}**

**}**

**if** **(**tmp **==** 0**)** **{**

output **<<** gradebook\_number **<<** " " **<<** name **<<** " " **<<** surname **<<** " " **<<** patronymic **<<** " " **<<** gen\_der **<<** " " **<<** day **<<** " " **<<** month **<<** " " **<<** year **<<** " " **<<** year\_of\_receipt **<<** " " **<<** faculty **<<** " " **<<** department **<<** " " **<<** group **<<** endl**;**

**}**

**}**

ifstream if\_rewrite**(**"Replaser.txt"**,** std**::**ios**::**app**);**

ofstream of\_rewrite**(**"Base.txt"**);**

count **-=** 1**;**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

if\_rewrite **>>** gradebook\_number **>>** name **>>** surname **>>** patronymic **>>** gen\_der **>>** day **>>** month **>>** year **>>** year\_of\_receipt **>>** faculty **>>** department **>>** group**;**

of\_rewrite **<<** gradebook\_number **<<** " " **<<** name **<<** " " **<<** surname **<<** " " **<<** patronymic **<<** " " **<<** gen\_der **<<** " " **<<** day **<<** " " **<<** month **<<** " " **<<** year **<<** " " **<<** year\_of\_receipt **<<** " " **<<** faculty **<<** " " **<<** department **<<** " " **<<** group **<<** endl**;**

**}**

cout **<<** "Студент Удален Из Списка" **<<** endl**;**

if\_rewrite**.**close**();**

of\_rewrite**.**close**();**

**input.**close**();**

output**.**close**();**

**}**

void Change\_Student**()** **{**

ifstream **input(**"Base.txt"**,** std**::**ios**::**app**);**

ofstream output**(**"Replaser.txt"**);**

cout **<<** endl **<<** "Введите Номер Студенческого Билета: " **<<** endl**;**

cin **>>** number**;**

clear\_all**();**

cout **<<** endl**;**

cout **<<** "Что Вы Хотите Изменить? " **<<** endl **<<** endl**;**

cout **<<** "[1] ФИО, Пол" **<<** endl**;**

cout **<<** "[2] {День, Месяц, Год} - Рождения и Год Поступления" **<<** endl**;**

cout **<<** "[3] Факультет, Кафедра, Группа" **<<** endl**;**

cout **<<** "[4] Номер Студенческого Билета" **<<** endl**;**

cout **<<** " > "**;**

command **=** protect\_change**();**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

**input** **>>** gradebook\_number **>>** name **>>** surname **>>** patronymic **>>** gen\_der **>>** day **>>** month **>>** year **>>** year\_of\_receipt **>>** faculty **>>** department **>>** group**;**

**int** tmp**(**1**);**

**for** **(int** i **=** 0**;** i **<** 7**;** i**++)** **{**

**if** **(**gradebook\_number**[**i**]** **!=** number**[**i**])** **{**

tmp **=** 0**;**

**break;**

**}**

**}**

**if** **(**tmp **==** 1**)** **{**

switch **(**command**)** **{**

case 1**:**

**{**

cout **<<** "Введите ФИО и Пол{M/F} Студента: " **<<** endl**;**

cin **>>** surname **>>** name **>>** patronymic **>>** gen\_der**;**

clear\_all**();**

**break;**

**}**

case 2**:**

**{**

cout **<<** "Введите {День, Месяц, Год} - Рождения и Год По-ступления: " **<<** endl**;**

day **=** protect\_day**();**

month **=** protect\_month**();**

year **=** protect\_year**();**

year\_of\_receipt **=** protect\_year**();**

**break;**

**}**

case 3**:**

**{**

cout **<<** "Введите Название Факультета, Кафедры, Группы" **<<** endl**;**

cin **>>** faculty **>>** department **>>** group**;**

clear\_all**();**

**break;**

**}**

case 4**:**

**{**

cout **<<** "Введите Новый Номер Студенческого Билета" **<<** endl**;**

cin **>>** gradebook\_number**;**

clear\_all**();**

**break;**

**}**

**}**

output **<<** gradebook\_number **<<** " " **<<** name **<<** " " **<<** surname **<<** " " **<<** patronymic **<<** " " **<<** gen\_der **<<** " " **<<** day **<<** " " **<<** month **<<** " " **<<** year **<<** " " **<<** year\_of\_receipt **<<** " " **<<** faculty **<<** " " **<<** department **<<** " " **<<** group **<<** endl**;**

**}**

**else** output **<<** gradebook\_number **<<** " " **<<** name **<<** " " **<<** surname **<<** " " **<<** patronymic **<<** " " **<<** gen\_der **<<** " " **<<** day **<<** " " **<<** month **<<** " " **<<** year **<<** " " **<<** year\_of\_receipt **<<** " " **<<** faculty **<<** " " **<<** department **<<** " " **<<** group **<<** endl**;**

**}**

ifstream if\_rewrite**(**"Replaser.txt"**,** std**::**ios**::**app**);**

ofstream of\_rewrite**(**"Base.txt"**);**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

if\_rewrite **>>** gradebook\_number **>>** name **>>** surname **>>** patronymic **>>** gen\_der **>>** day **>>** month **>>** year **>>** year\_of\_receipt **>>** faculty **>>** department **>>** group**;**

of\_rewrite **<<** gradebook\_number **<<** " " **<<** name **<<** " " **<<** surname **<<** " " **<<** patronymic **<<** " " **<<** gen\_der **<<** " " **<<** day **<<** " " **<<** month **<<** " " **<<** year **<<** " " **<<** year\_of\_receipt **<<** " " **<<** faculty **<<** " " **<<** department **<<** " " **<<** group **<<** endl**;**

**}**

cout **<<** "Студент Изменен" **<<** endl**;**

if\_rewrite**.**close**();**

of\_rewrite**.**close**();**

**input.**close**();**

output**.**close**();**

**}**

void Find\_Student**()** **{**

ifstream **input(**"Base.txt"**,** std**::**ios**::**app**);**

**int** C\_tmp**(**0**);**

cout **<<** endl **<<** "Введите Номер Студенческого Билета: " **<<** endl**;**

cin **>>** number**;**

clear\_all**();**

**for** **(int** i **=** 0**;** i **<** count**;** i**++)** **{**

**input** **>>** gradebook\_number **>>** name **>>** surname **>>** patronymic **>>** gen\_der **>>** day **>>** month **>>** year **>>** year\_of\_receipt **>>** faculty **>>** department **>>** group**;**

**int** tmp**(**1**);**

**for** **(int** i **=** 0**;** i **<** 7**;** i**++)** **{**

**if** **(**gradebook\_number**[**i**]** **!=** number**[**i**])** **{**

tmp **=** 0**;**

**break;**

**}**

**}**

**if** **(**tmp **==** 1**)** **{**

cout **<<** endl **<<** "Имя Студента: "**;**

cout **<<** name**;**

cout **<<** endl **<<** "Фамилия Студента: "**;**

cout **<<** surname**;**

cout **<<** endl **<<** "Отчество Студента: "**;**

cout **<<** patronymic**;**

cout **<<** endl **<<** "Пол Студента: "**;**

cout **<<** gen\_der**;**

cout **<<** endl **<<** "Дата Рождения Студента {Д,М,Г}: "**;**

cout **<<** day **<<** " " **<<** month **<<** " " **<<** year**;**

cout **<<** endl **<<** "Дата Поступления Студента: "**;**

cout **<<** year\_of\_receipt**;**

cout **<<** endl **<<** "Факультет Студента: "**;**

cout **<<** faculty**;**

cout **<<** endl **<<** "Кафедра Студента: "**;**

cout **<<** department**;**

cout **<<** endl **<<** "Группа Студента: "**;**

cout **<<** group**;**

cout **<<** endl **<<** "Номер Студенческого Билета: "**;**

cout **<<** gradebook\_number**;**

cout **<<** endl **<<** endl**;**

C\_tmp **=** 1**;**

**break;**

**}**

**}**

**input.**close**();**

**if** **(**C\_tmp **==** 0**)** cout **<<** "Студент Не Найден" **<<** endl **<<** endl**;**

**}**

void clear\_all**()** **{**

cin**.**seekg**(**0**,** ios**::**end**);**

cin**.**clear**();**

**}**

**~**Menu**()** **{**

**}**

**};**

**int** main**()** **{**

Menu menu**;**

menu**.**welcome**();**

**return** 0**;**

**}**