Министерство науки и высшего образования   
Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение высшего образования  
«Рязанский государственный радиотехнический   
университет имени В. Ф. Уткина»  
Кафедра вычислительной и прикладной математики

**Пояснительная записка  
к курсовому проекту по дисциплине**«Алгоритмические языки и программирование»   
**на тему**“Работа с базой данных”

Выполнил: студент группы 344  
Гомелаури Е. А.  
Проверил: доцент  
Дмитриева Т. А.

Рязань 2024

Задание

Введение

В современном мире информационные технологии играют ключевую роль в различных сферах жизни. Одним из важнейших аспектов этой области является работа с базами данных, которые позволяют эффективно хранить, обрабатывать и анализировать большие объемы информации. Данная курсовая работа посвящена разработке и внедрению базы данных, что представляет собой актуальную и значимую задачу в условиях стремительного развития цифровых технологий.

В результате выполнения данной курсовой работы была разработана и внедрена база данных, полностью удовлетворяющая поставленным требованиям. Система прошла успешное тестирование и была оценена на предмет производительности и надежности. Разработанная документация позволит пользователям и администраторам эффективно взаимодействовать с базой данных и поддерживать её в актуальном состоянии.

Таким образом, данная работа демонстрирует возможность успешного решения задач по разработке и внедрению баз данных, что подтверждает её практическую значимость и актуальность в современных условиях.

1. Анализ задания и математическая постановка задачи

Задание состоит в разработке программы для работы с базой данных учеников класса, представленной в виде словаря. Программа должна уметь дополнять базу новыми записями и выполнять поиск номеров телефонов по фамилии с циклическим запросом продолжения поиска. Также необходимо реализовать систему меню для управления базой данных.

Для реализации программы необходимы предусмотреть следующие компоненты:

1. Пакет модулей для работы с базой данных
   * Модуль с подпрограммами для изменения, удаления и добавления записей в базу данных
   * Модуль с подпрограммами для просмотра данных в базе и поиска необходимой информации (поиск номера телефона по фамилии)
2. Пакет модулей для работы с файлом
   * Модуль с подпрограммой для чтения данных из файла
   * Модуль с подпрограммой для перезаписи данных в файл

Все компоненты подключаются к основной программе, в которой реализовано меню для выбора действий над базой данных.

1. Разработка схем алгоритма и ее описание
   1. Схема модуля основной программы.
      1. Основная программа

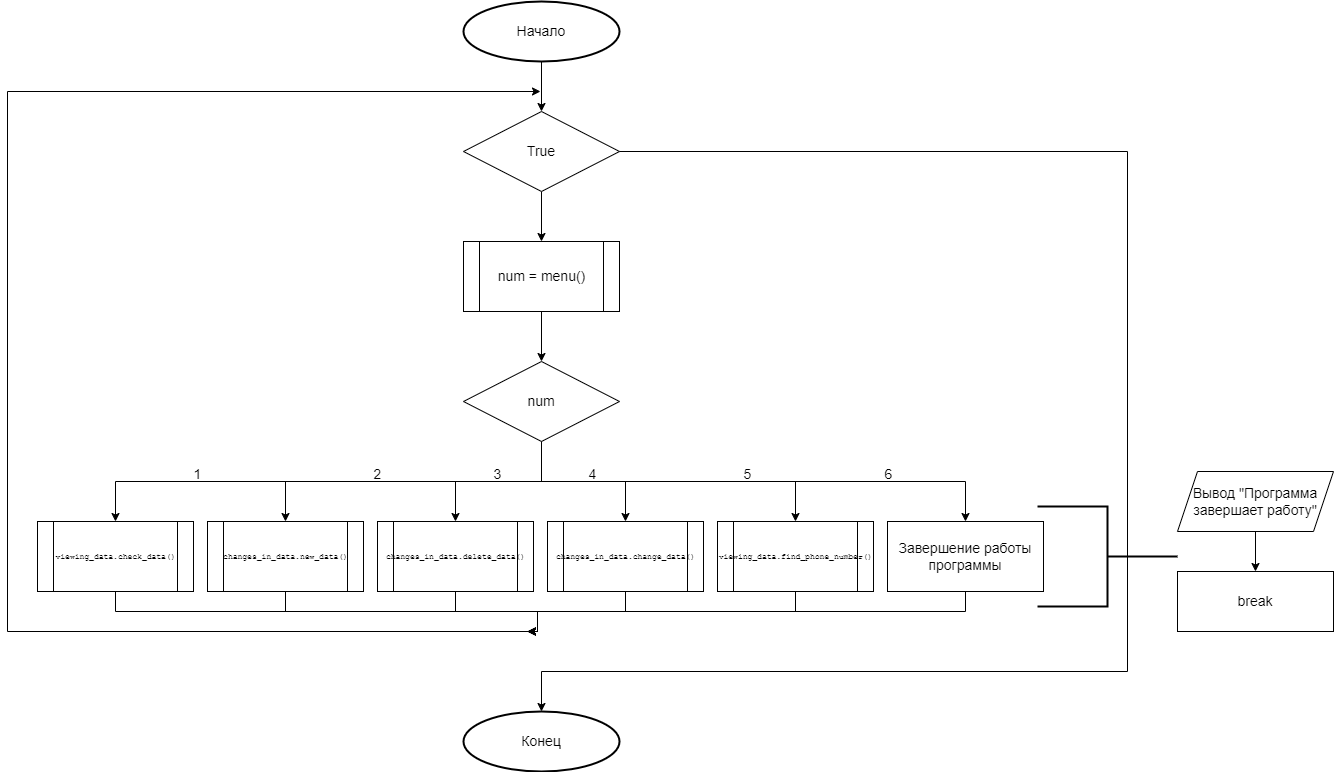


Рисунок 1 – Блок-схема основной программы

* + 1. Функция menu

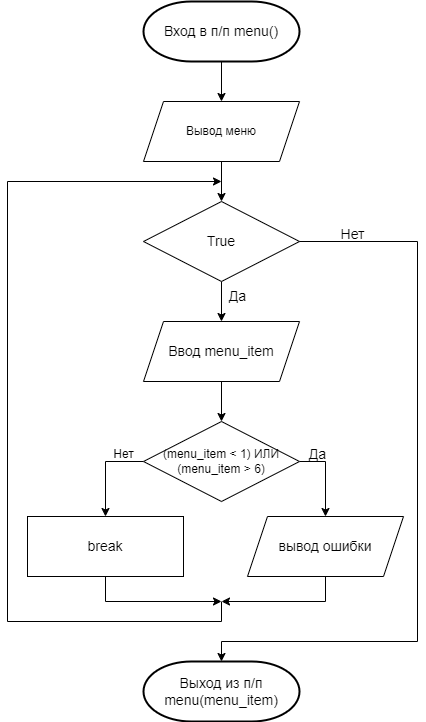


Рисунок 2 – Блок-схема подпрограммы menu()

* + 1. Модуль input

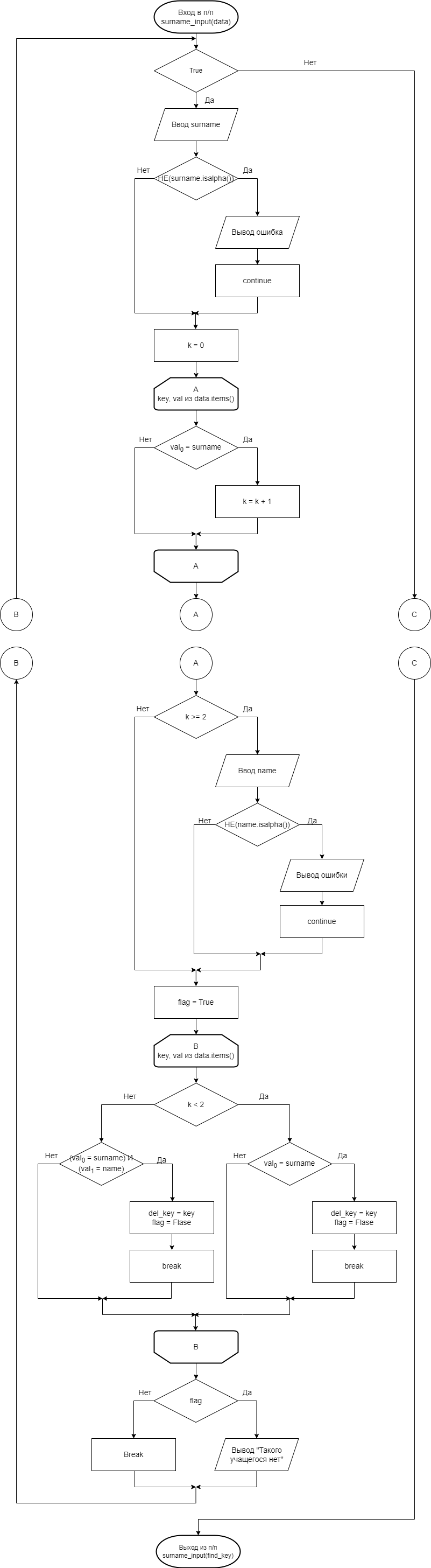


Рисунок 3 – Блок-схема подпрограммы input\_surname()

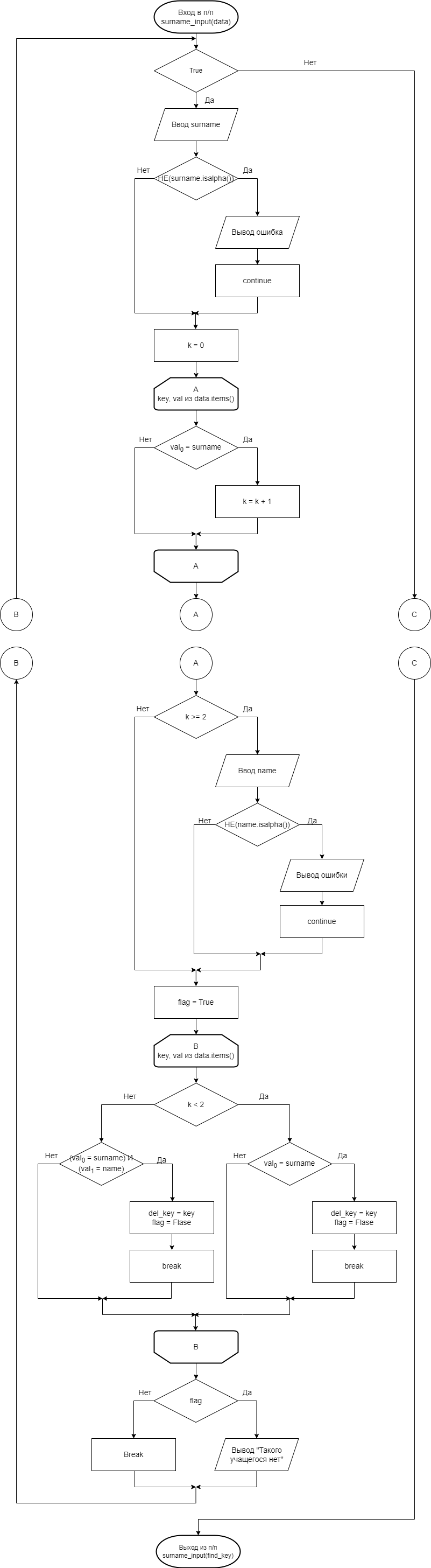


Рисунок 4 – Блок-схема подпрограммы input\_surname()

* 1. Схемы пакета модулей для работы с базой данных
     1. Модуль viewing\_data

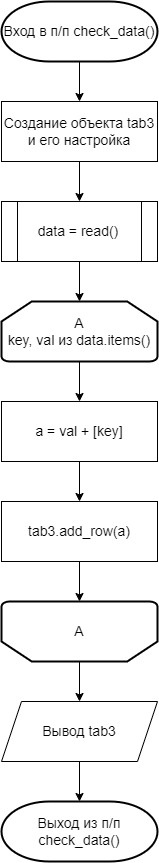


Рисунок 5 – Блок-схема подпрограммы check\_data()

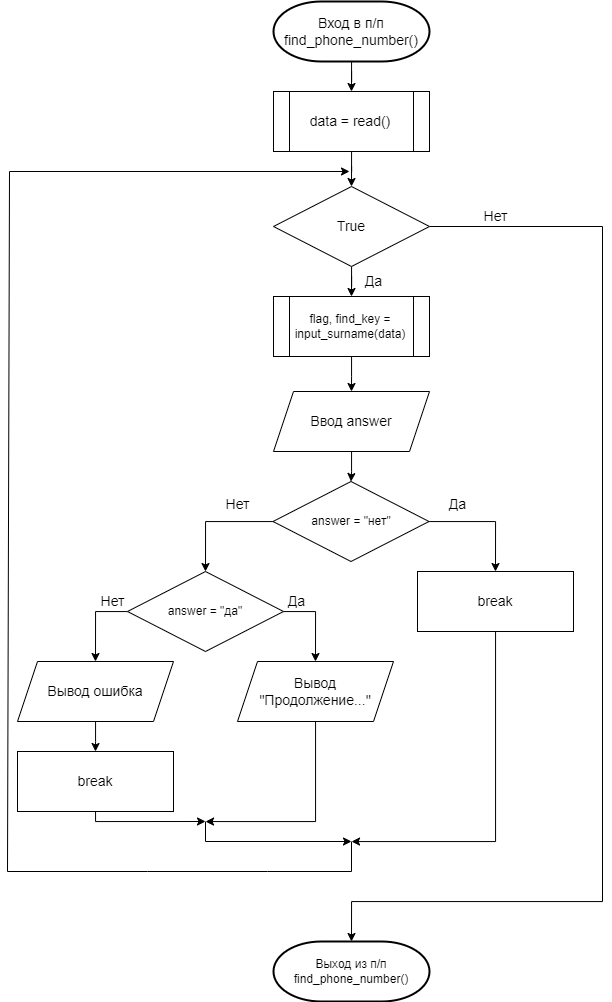


Рисунок 6 – Блок-схема функции find\_phone\_number()

* + 1. Модуль changes\_in\_data

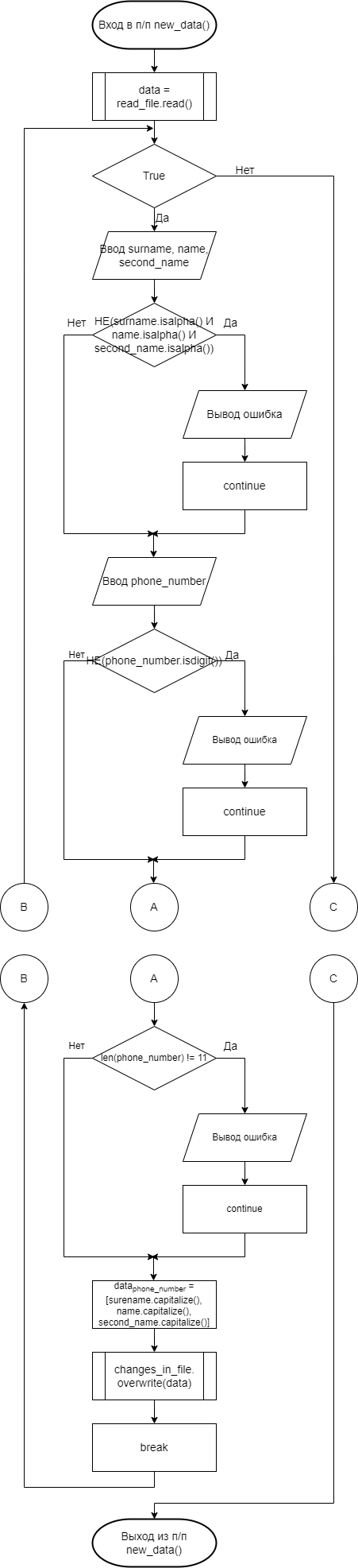


Рисунок 7 – Блок-схема функции new\_data()

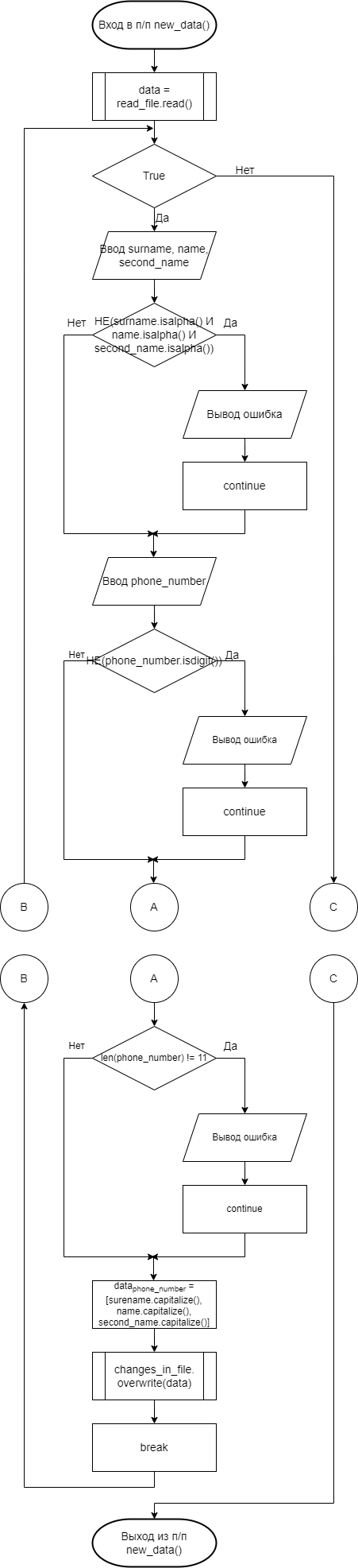


Рисунок 8 – Блок-схема функции new\_data()

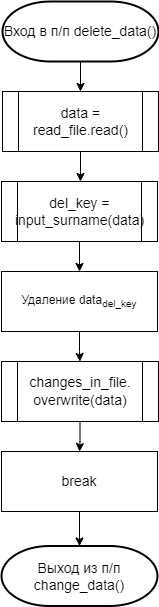
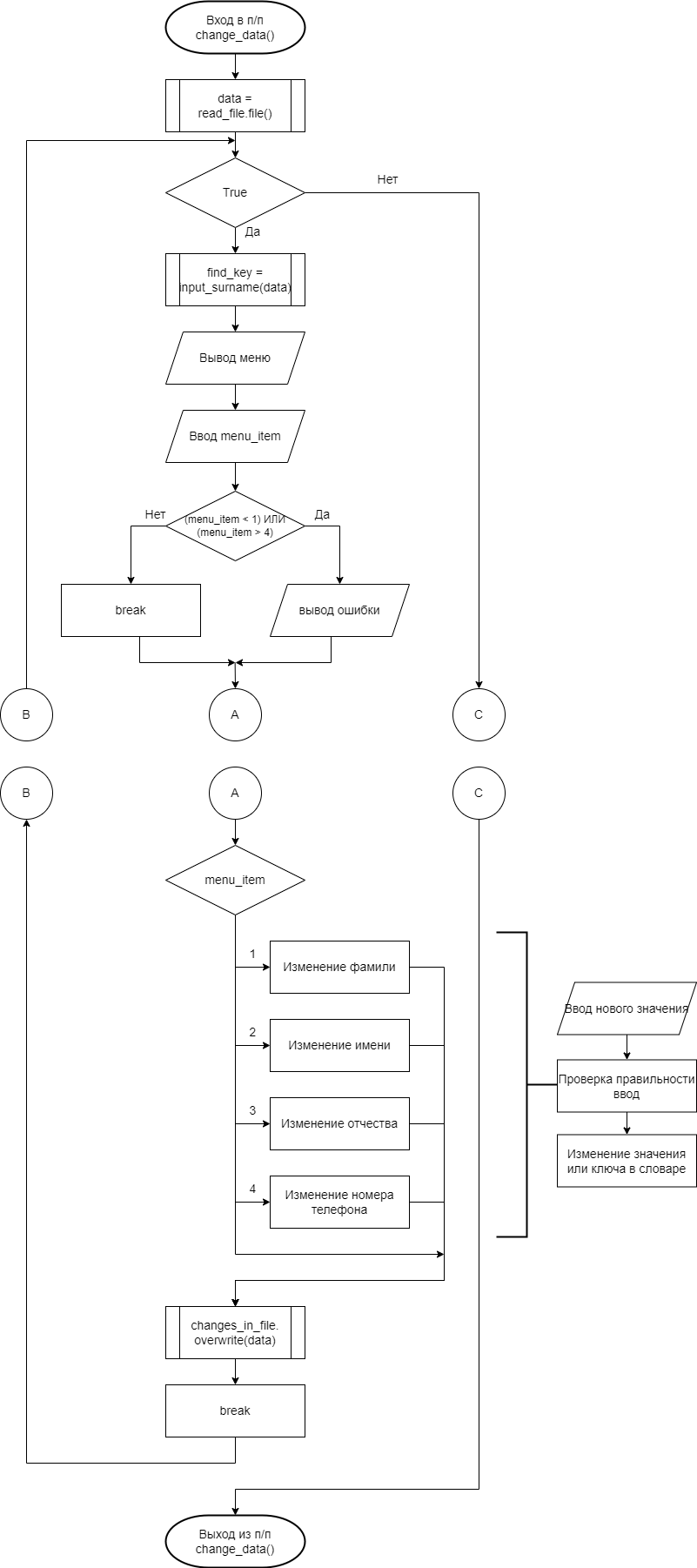


Рисунок 9 – Блок-схема функции delete\_data()



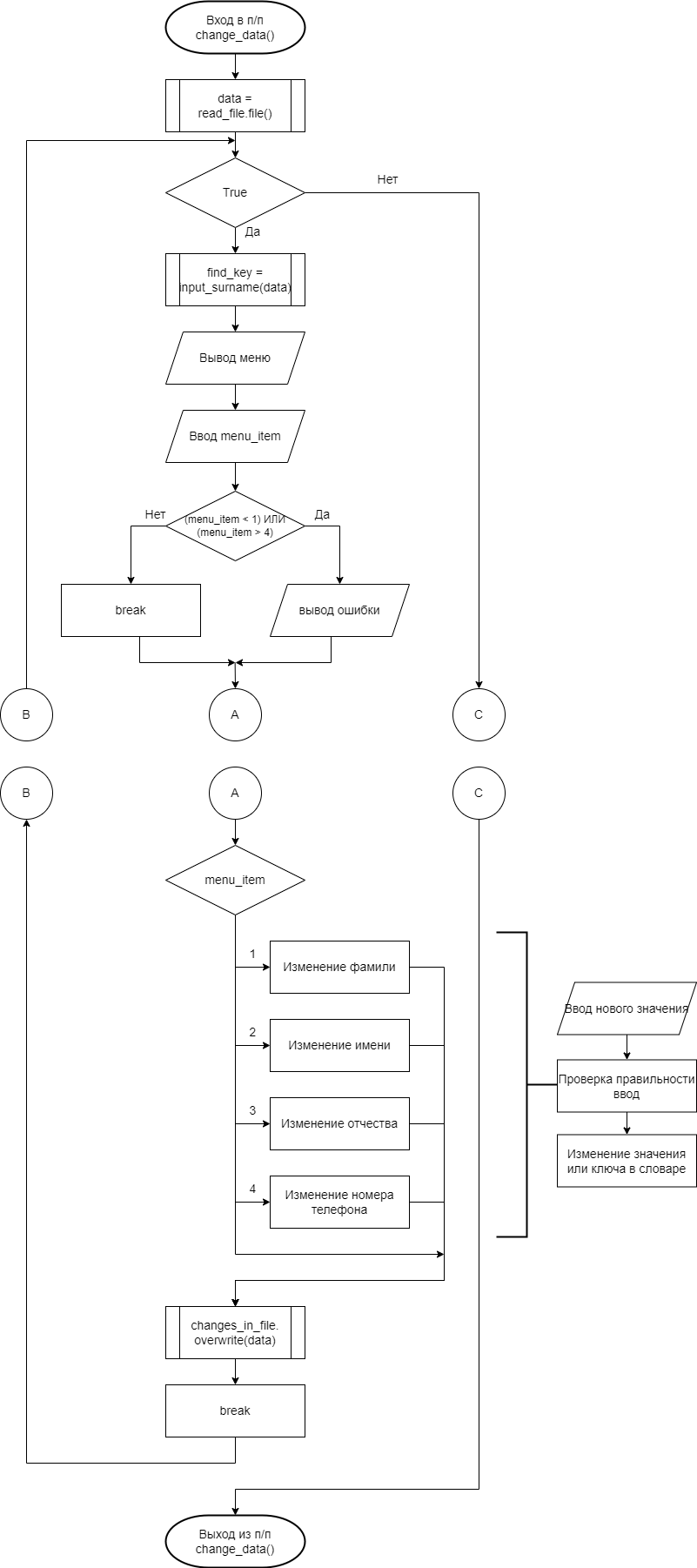


Рисунок 10 – Блок-схема функции change\_data()

* 1. Схемы пакета модулей для работы с файлом
     1. Модуль read\_file

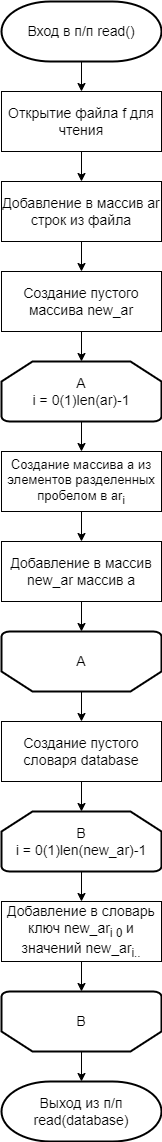


Рисунок 11 – Блок-схема функции read()

* + 1. Модуль changes\_in\_file

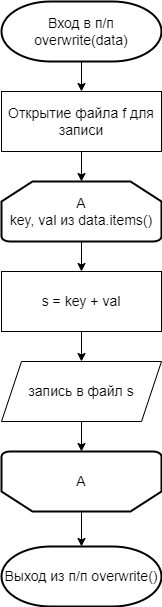


Рисунок 12 – Блок-схема функции overwrite()

1. Разработка программы и ее описание

Интерфейс реализован с помощью консоли. Взаимодействию с меню с помощью клавиатуры.

Для разработки программы необходимо выяснить как должна выглядеть база данных. Пример приведен в таблице 1.

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| Номер телефона | Фамилия | Имя | Отчество |
| Номер телефона 1 | Фамилия 1 | Имя 1 | Отчество 1 |
| Номер телефона 2 | Фамилия 2 | Имя 2 | Отчество 2 |
| … | … | … | … |
| Номер телефона n | Фамилия n | Имя n | Отчество n |

База данных хранится в файле, находящемся в одном каталоге вместе с программой.

3.1. Библиотека модулей

В программе представлены два пакета модулей и отдельный модуль:

1. Пакет модулей working\_with\_data
   1. changes\_in\_data – модуль, в котором содержаться подпрограммы для изменения, добавления или удаления данных из базы (блок-схема представлена на рисунках 7 -10);
   2. viewing\_data – модуль, в котором содержится подпрограмма для просмотра базы данных и подпрограмма для поиска нужной информации(блок-схемы представлены на рисунках 5-6);
2. Пакет модулей working\_with\_file
   1. read\_file – модуль, в котором содержится подпрограмма для считывания данных из файла и создания из них словаря (блок-схема представлена на рисунке 11);
   2. changes\_in\_file – модуль, в котором содержится подпрограмма для перезаписи данных в файле, т. е. старые данные стираются и добавляется измененный словарь (блок-схема представлена на рисунке 12);
3. input – модуль, в котором содержится подпрограмма для поиска ключ по значению (блок-схема представлена на рисунка 3-4)
4. Инструкция по использованию разработанной программой

Для открытия программы необходимо запустить файл «main.py».

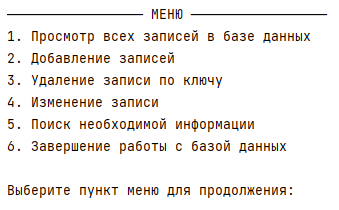


Рисунок 13 – Меню функция

После открытия пользователь увидит меню, в котором можно будет выбрать нужную функцию. Для этого пользователю нужно ввести цифру и нажать кнопку Enter (рисунок 13). Если пользователь введет символ отличный от пункта меню, программа сообщит об ошибке (рисунок 14).



Рисунок 14 – Сообщение об ошибке

При выборе пункта меню «Просмотр всех записей в базе данных» выводится таблица с данными, которые содержатся в файле (рисунок 15).

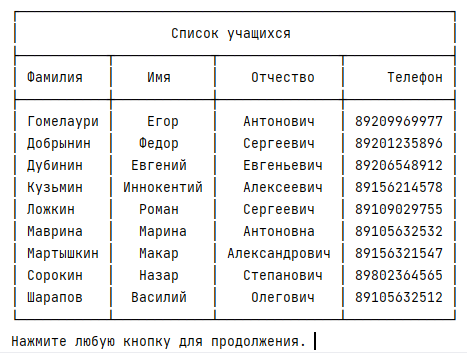


Рисунок 15 – Таблица с данными из файла

При выборе пункта «Добавление записей», программа запрашивает все необходимые данные. При вводе некорректных данных программа сообщает об ошибке (рисунки 16-17).

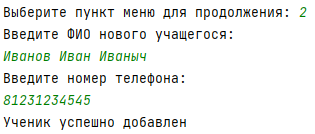


Рисунок 16 – Добавление записи

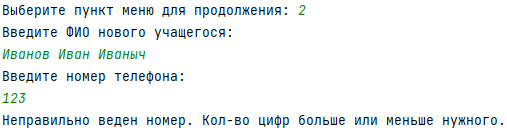


Рисунок 17 – Сообщение об ошибке при неправильных вводимых данных

При выборе пункта «Удаление записей», программа запрашивает фамилию, если учащихся с одинаковой фамилией несколько, то программа запрашивает имя (рисунок 18).



Рисунок 18 – Удаление записи

При выборе пункта «Изменение записи», программа запрашивает фамилию учащегося, данные которого нужно изменить. Затем выводиться меню, в котором можно выбрать, что конкретно нужно изменить. После выбора пункта меню программа запрашивает новые данные и обновляет запись (рисунок 19).

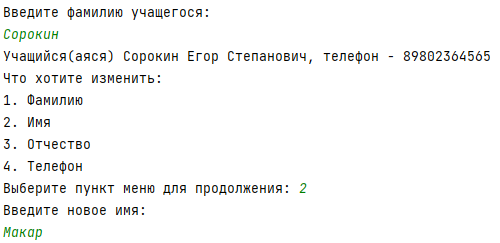


Рисунок 19 – Изменение записи

При выборе пункта «Поиск необходимой информации», программа запрашивает фамилию и выводит номер телефона. После программа предлагает продолжить поиск (рисунок 20).

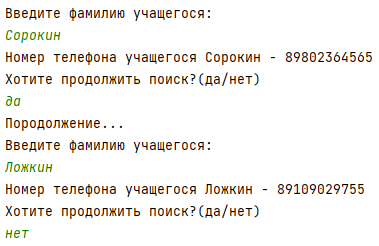


Рисунок 20 – Поиск номера телефона по фамилии

При выборе пункта «Завершение работы с базой данных» программа выводит сообщение о завершении (рисунок 21).



Рисунок 21 – Завершение работы

1. Отладка программы

Компиляция программы прошла без ошибок.

1. Решение контрольных примеров и проверка правильности функционирования программы

Для проверки правильности функционирования программы, воспользуемся каждой функцией, а именно:

1. Добавление записи
2. Изменение записи:
   1. Изменение фамилии
   2. Изменение имени
   3. Изменение отчества
   4. Изменение номера телефона
3. Поиск информации
4. Удаление

В программе предусмотрены случаи, когда пользователь может ввести некорректные данные, в данном случае программа выведет сообщение об ошибке и предоставит возможность заново ввести данные.

6.1. Тестирование

1. При добавление нового учащегося его запись будет находиться в списке по алфавиту (рисунки 22 - 23).

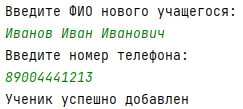


Рисунок 22



Рисунок 23

1. Изменение записи
   1. Изменение фамилии (рисунки 24-25)

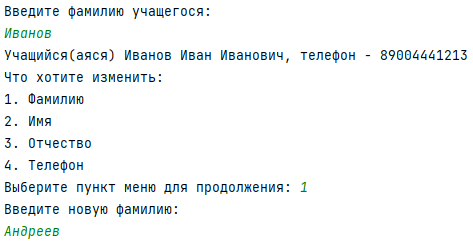


Рисунок 24



Рисунок 25

* 1. Изменение имени (рисунки 26-27)

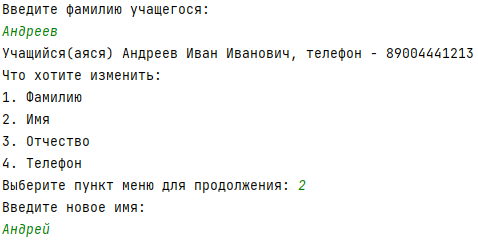


Рисунок 26



Рисунок 27

* 1. Изменение отчества (рисунки 28-29)

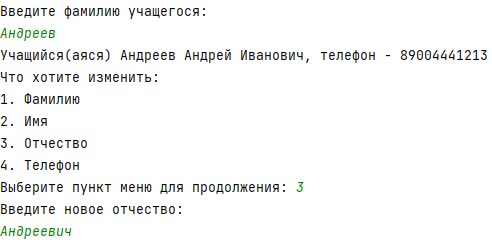


Рисунок 28



Рисунок 29

* 1. Изменение телефона (рисунки 30-31)

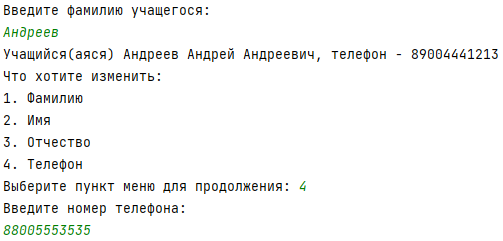


Рисунок 30



Рисунок 31

1. Поиск информации (рисунок 32)

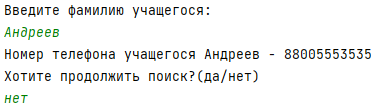


Рисунок 32

1. Удаление записи (рисунки 33-34)

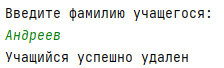


Рисунок 33



Рисунок 34

Решение контрольного примера показало, что программа работает верно.

Заключение

В ходе работы были рассмотрены основные концепции языка, такие как переменные, операторы, условные операторы, циклы, процедуры и функции. Также были изучены некоторые особенности языка, такие как работа с типами данных и работа с файлами. Все эти знания были применены в написании этой курсовой работе.

Мы использовали такие методы как анализ и постановка задачи, разработка блок-схем используемых алгоритмов, реализация и тестирование программы.

Приложение А – Файл «main.py»

from working\_with\_database import \*  
  
  
def menu() -> int:  
 *"""Функция вывода меню  
 Выходные данные:  
 menu\_item - номер пункта меню"""  
  
 # Вывод меню* print(' МЕНЮ '.center(40, "─"))  
 print('1. Просмотр всех записей в базе данных')  
 print('2. Добавление записей')  
 print('3. Удаление записи')  
 print('4. Изменение записи')  
 print('5. Поиск необходимой информации')  
 print('6. Завершение работы с базой данных\n')  
  
 *# Ввод значения и проверка корректности введенного значения* while True:  
 try:  
 menu\_item = int(input('Выберите пункт меню для продолжения: '))  
  
 if (menu\_item < 1) or (menu\_item > 6):  
 print('Что-то пошло не так =(')  
 print('Некорректно введен пункт меню (значение не попадает в диапозон от 1 до 6). Попробуйте снова.')  
 else:  
 break  
  
 except ValueError:  
 print('Что-то пошло не так =(')  
 print('Неверно введен пункт меню (необходимо ввести одно целое число). Попробуйте снова')  
  
 return menu\_item  
  
  
while True:  
 num = menu()  
 match num:  
 case 1:  
 viewing\_data.check\_data()  
 case 2:  
 changes\_in\_data.new\_data()  
 case 3:  
 changes\_in\_data.delete\_data()  
 case 4:  
 changes\_in\_data.change\_data()  
 case 5:  
 viewing\_data.find\_phone\_number()  
 case 6:  
 print('Программа завершает работу')  
 break

Приложение Б – файл «input.py»

def input\_surname(data):  
  
 while True:  
 print('Введите фамилию учащегося: ')  
 surname = input().capitalize()  
  
 *# Проверка на ввод букв* if not surname.isalpha():  
 print('Что-то пошло не так =(')  
 print('Скорее всего вместо букв были использованы цифры.')  
 continue  
  
 *# Проверка на кол-во учащихся с одинаковой фамилией* k = 0  
 for val in data.values():  
 if val[0] == surname:  
 k += 1  
  
 if k >= 2:  
 print('Учащихся с такой фамилие несколько. Уточните Имя')  
 name = input().capitalize()  
 if not name.isalpha():  
 print('Что-то пошло не так =(')  
 print('Скорее всего вместо букв были использованы цифры.')  
 continue  
  
 *# Поиск ключа по значению* flag = True  
 for key, val in data.items():  
 if k < 2:  
 if val[0] == surname:  
 find\_key = key  
 flag = False  
 break  
 else:  
 if val[0] == surname and val[1] == name:  
 find\_key = key  
 flag = False  
 break  
  
 if flag:  
 print('Такого учащегося нет')  
 continue  
 else:  
 break  
  
 return find\_key

Приложение В – файл «changes\_in\_data.py»

from working\_with\_file import \*  
from input import input\_surname  
  
  
def new\_data():  
 *"""Функция добавления новой записи в базу данных"""* data = read\_file.read()  
  
 while True:  
 try:  
 print('Введите ФИО нового учащегося:')  
 surname, name, second\_name = input().split()  
 except ValueError:  
 print('Что-то пошло не так =(')  
 print('Введите фамилию, имя и отчество через пробел.')  
 continue  
  
 *# Проверка на ввод букв* if not (surname.isalpha() and name.isalpha() and second\_name.isalpha()):  
 print('Что-то пошло не так =(')  
 print('Скорее всего вместо букв были использованы цифры.')  
 continue  
  
 print('Введите номер телефона: ')  
 phone\_number = input()  
  
 *# Проверка на ввод цифр* if not phone\_number.isdigit():  
 print('Что-то пошло не так =(')  
 print('Скорее всего вместо цифр были использованы буквы.')  
 continue  
  
 *# Проверка на кол-во цифр в номере* if len(phone\_number) != 11:  
 print('Неправильно веден номер. Кол-во цифр больше или меньше нужного.')  
 continue  
  
 data[phone\_number] = [surname.capitalize(), name.capitalize(), second\_name.capitalize()]  
 changes\_in\_file.overwrite(data)  
 print('Ученик успешно добавлен')  
 break  
  
  
def delete\_data():  
 *"""Функция удаления элемента из базы данных"""* data = read\_file.read()  
  
 del\_key = input\_surname(data)  
  
 *# Удаление учащегося по ключу* del data[del\_key]  
 changes\_in\_file.overwrite(data)  
 print('Учащийся успешно удален')  
  
  
def change\_data():  
 *"""Функция изменения элемента в базе данных"""* data = read\_file.read()  
  
 while True:  
  
 find\_key = input\_surname(data)  
  
 *# Вывод меню* print(f'Учащийся(аяся) {data[find\_key][0] + " " + data[find\_key][1] + " " + data[find\_key][2]}, телефон - {find\_key}')  
 print(f'Что хотите изменить: ')  
 print('1. Фамилию')  
 print('2. Имя')  
 print('3. Отчество')  
 print('4. Телефон')  
  
 *# Ввод и проверка на правильность ввода* try:  
 menu\_item = int(input('Выберите пункт меню для продолжения: '))  
  
 if (menu\_item < 1) or (menu\_item > 4):  
 print('Что-то пошло не так =(')  
 print('Некорректно введен пункт меню (значение не попадает в диапозон от 1 до 4). Попробуйте снова.')  
 continue  
  
 except ValueError:  
 print('Что-то пошло не так =(')  
 print('Неверно введен пункт меню (необходимо ввести одно целое число). Попробуйте снова')  
 continue  
  
 match menu\_item:  
 case 1:  
 *# Изменение фамилии* print('Введите новую фамилию:')  
 surname = input().capitalize()  
  
 if not surname.isalpha():  
 print('Что- то пошло не так =(')  
 print('Скорее всего вместо букв были использованы цифры.')  
 continue  
 data[find\_key][0] = surname  
  
 case 2:  
 *# Изменение имени* print('Введите новое имя:')  
 name = input().capitalize()  
  
 if not name.isalpha():  
 print('Что- то пошло не так =(')  
 print('Скорее всего вместо букв были использованы цифры.')  
 continue  
 data[find\_key][1] = name  
  
 case 3:  
 *# Изменение отчества* print('Введите новое отчество:')  
 second\_name = input().capitalize()  
  
 if not second\_name.isalpha():  
 print('Что- то пошло не так =(')  
 print('Скорее всего вместо букв были использованы цифры.')  
 continue  
 data[find\_key][2] = second\_name  
  
 case 4:  
 *# Изменение номера телефона* print('Введите номер телефона: ')  
 phone\_number = input()  
  
 *# Проверка на ввод цифр* if not phone\_number.isdigit():  
 print('Что-то пошло не так =(')  
 print('Скорее всего вместо цифр были использованы буквы.')  
 continue  
  
 *# Проверка на кол-во цифр в номере* if len(phone\_number) != 11:  
 print('Неправильно веден номер. Кол-во цифр больше или меньше нужного.')  
 continue  
 data[phone\_number] = data.pop(find\_key)  
  
 changes\_in\_file.overwrite(data)  
 break

Приложение Г – файл «viewing\_data.py»

from prettytable import PrettyTable  
  
from working\_with\_file.read\_file import read  
from input import input\_surname  
  
  
def check\_data():  
 *"""Функция вывода базы данных"""  
  
 # Настройка таблицы* tab3 = PrettyTable(title='Список учащихся', border=True, header=True,  
 vertical\_char='│',  
 horizontal\_char='─',  
 junction\_char='┼',  
 left\_junction\_char='├',  
 right\_junction\_char='┤',  
 top\_left\_junction\_char='┌',  
 top\_right\_junction\_char='┐',  
 top\_junction\_char='┬',  
 bottom\_junction\_char='┴',  
 bottom\_right\_junction\_char='┘',  
 bottom\_left\_junction\_char='└',  
 start=0,  
 )  
  
 tab3.field\_names = ['Фамилия', 'Имя', 'Отчество', 'Телефон']  
 tab3.align['Фамилия'] = 'l'  
 tab3.align['Имя'] = 'с'  
 tab3.align['Отчество'] = 'с'  
 tab3.align['Телефон'] = 'r'  
 tab3.sortby = 'Фамилия'  
  
 data = read()  
  
 *# Заполнение таблицы значениями из словаря* for key, val in data.items():  
 a = val + [key]  
 tab3.add\_row(a)  
 print(tab3)  
  
 *# Возврат в меню* input('Нажмите любую кнопку для продолжения. ')  
  
  
def find\_phone\_number():  
 *"""Функция поиска номера телефона по фамилии"""* data = read()  
  
 while True:  
  
 find\_key = input\_surname(data)  
  
 print(f'Номер телефона учащегося {data[find\_key][0]} - {find\_key}')  
  
 print('Хотите продолжить поиск?(да/нет)')  
 answer = input()  
  
 if answer == 'нет':  
 break  
 elif answer == 'да':  
 print('Породолжение...')  
 else:  
 print('Что- то пошло не так =(')  
 print('Ответ неодназначен. Возврат в меню.')  
 break

Приложение Д – файл «changes\_in\_file.py»

def overwrite(database):  
 *"""Функция перезаписи данных в файле  
 Входные данные:  
 database - база данных"""* file\_name = 'C:/Users/winre/PycharmProjects/TelBook/data.txt'  
  
 with open(file\_name, 'w') as f:  
 for key, val in database.items():  
 s = key + '\t' + '\t'.join(val) + '\n'  
 f.write(s)

Приложение Е – файл «read\_file.py»

def read():  
 *"""Функция считывающая данные из файлы и преобразующая их в словарь  
 Выходные данные:  
 database - база данных """* file\_name = 'C:/Users/winre/PycharmProjects/TelBook/data.txt'  
  
 with open(file\_name, 'r') as f:  
 ar = f.readlines()  
  
 *# Разбиение строк на массивы* new\_ar = []  
 for i in range(len(ar)):  
 a = ar[i].split()  
 new\_ar.append([el.rstrip() for el in a])  
  
 *# Составление словаря из полученных массивов* database = {}  
 for i in range(len(new\_ar)):  
 database[new\_ar[i][0]] = new\_ar[i][1:]  
  
 return database