|  |  |  |
| --- | --- | --- |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное автономное  образовательное учреждение высшего образования  «Пермский государственный национальный  исследовательский университет» | | |
|  | Институт компьютерных наук и технологий | |
| **ОТЧЁТ**  по индивидуальной работе №1  по дисциплине «Языки программирования»  Вариант 4 | | |
|  | | Работу выполнил  студент группы ПМИ-9,10-2023 1 курса  Кулаков М.И.  «23» января 2024 г. |
| Работу проверил  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Фамилия И.О.  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г. |
| Пермь 2023 | | |

СОДЕРЖАНИЕ

[Постановка задачи 3](#_Toc153130027)

[Алгоритм решения 3](#_Toc153130028)

[Тестирование 3](#_Toc153130029)

[Код программы 3](#_Toc153130030)

[Инструкция по применению стилей и оформлению работы 4](#_Toc153130031)

# Постановка задачи

Создать файл записей, в котором хранится информация о владельцах автомобилей:

1. фамилия,

2. имя,

3. отчество,

4. регистрационный номер автомобиля (пример формата номера – К392НР)

5. марка автомобиля,

6. объём двигателя,

7. год выпуска

8. год постановки на учёт

Разработать и реализовать программу "Учёт в ГИБДД",

которая считывает исходную информацию и позволяет на основе неё создавать следующие отчёты:

1. Полный список всех автомобилистов, который будет отсортирован следующему ключу: год постановки на учёт (по убыванию) + фамилия владельца (по возрастанию).

2. Список всех владельцев автомобилей указанной марки (вводится с клавиатуры), отсортированный по следующему ключу:

год выпуска (по убыванию) + объём двигателя (по возрастанию) + фамилия (по возрастанию).

3. Список всех владельцев автомобилей с годом выпуска ранее заданного (вводится с клавиатуры), отсортированный по следующему ключу:

год выпуска (по убыванию) + марка автомобиля (по возрастанию).

Создать базу автовладельцев, включающую не менее 25 записей и на основе неё сформировать все указанные списки (можно использовать вложенные списки и словари) и на основе неё сформировать все указанные списки. Взаимодействие с базой данных обеспечивается отдельными функциями – добавление/удаление/изменение записи,

вывода на экран всей базы и отдельно сформированных списков. Интерфейс пользователя удобнее организовать в виде меню с выбором соответствующего действия. В программе обязательно наличие дружественного интерфейса и защиты от некорректного пользовательского ввода. База должна содержать такие записи, чтобы во всех списках явно прослеживался заданный вид сортировки по всем ключам. Для сортировки записей будет использоваться сортировка обменом

# Алгоритм решения

1. Создание файла, содержащего в себе не менее 25 записей с информацией о водителях.

2. Создание функции first\_check для проверки корректности данных файла с записями

для дальнейшей работы программы с ними.

3. Создание функции main\_menu для взаимодействия с пользователем и переходом к работе с отчётами, работе с данными файла и функцией выхода.  
  
4. Создание функции reports для выбора нужного нам отчёта.

5. Создание функции report\_1 для формирования отчёта по этому условию:

(1. Полный список всех автомобилистов, который будет отсортирован следующему ключу: год постановки на учёт (по убыванию) + фамилия владельца (по возрастанию).)

6. Создание функции report\_2 для формирования отчёта по этому условию:

(Список всех владельцев автомобилей указанной марки (вводится с клавиатуры), отсортированный по следующему ключу:

год выпуска (по убыванию) + объём двигателя (по возрастанию) + фамилия (по возрастанию).)

7. Создание функции report\_3 для формирования отчёта по этому условию:  
 (3. Список всех владельцев автомобилей с годом выпуска ранее заданного (вводится с клавиатуры), отсортированный по следующему ключу:

год выпуска (по убыванию) + марка автомобиля (по возрастанию).)

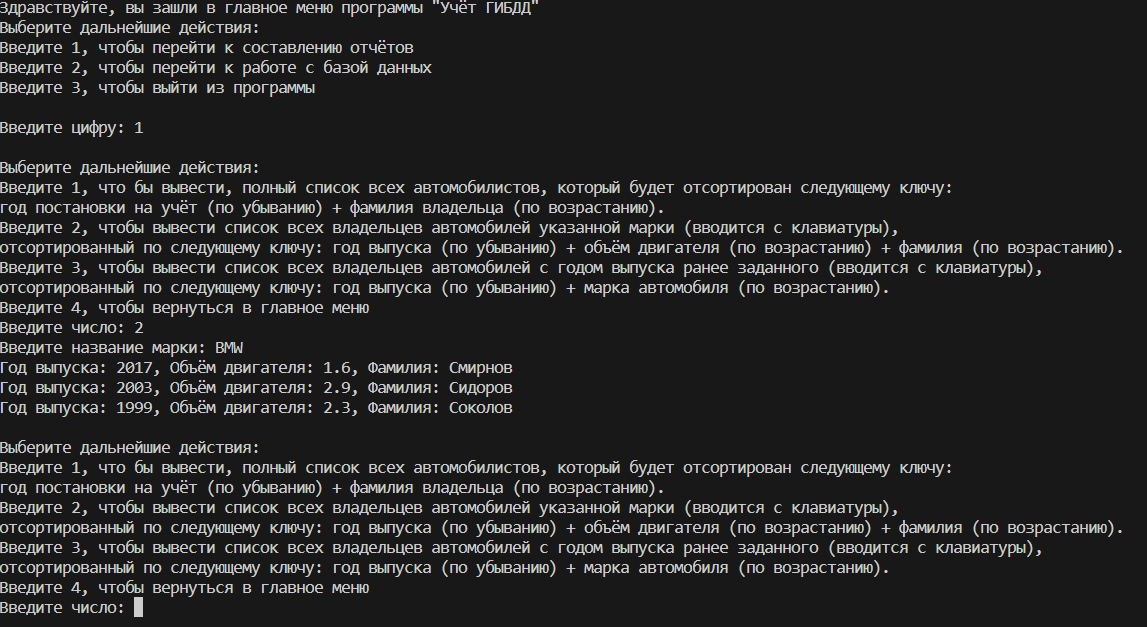
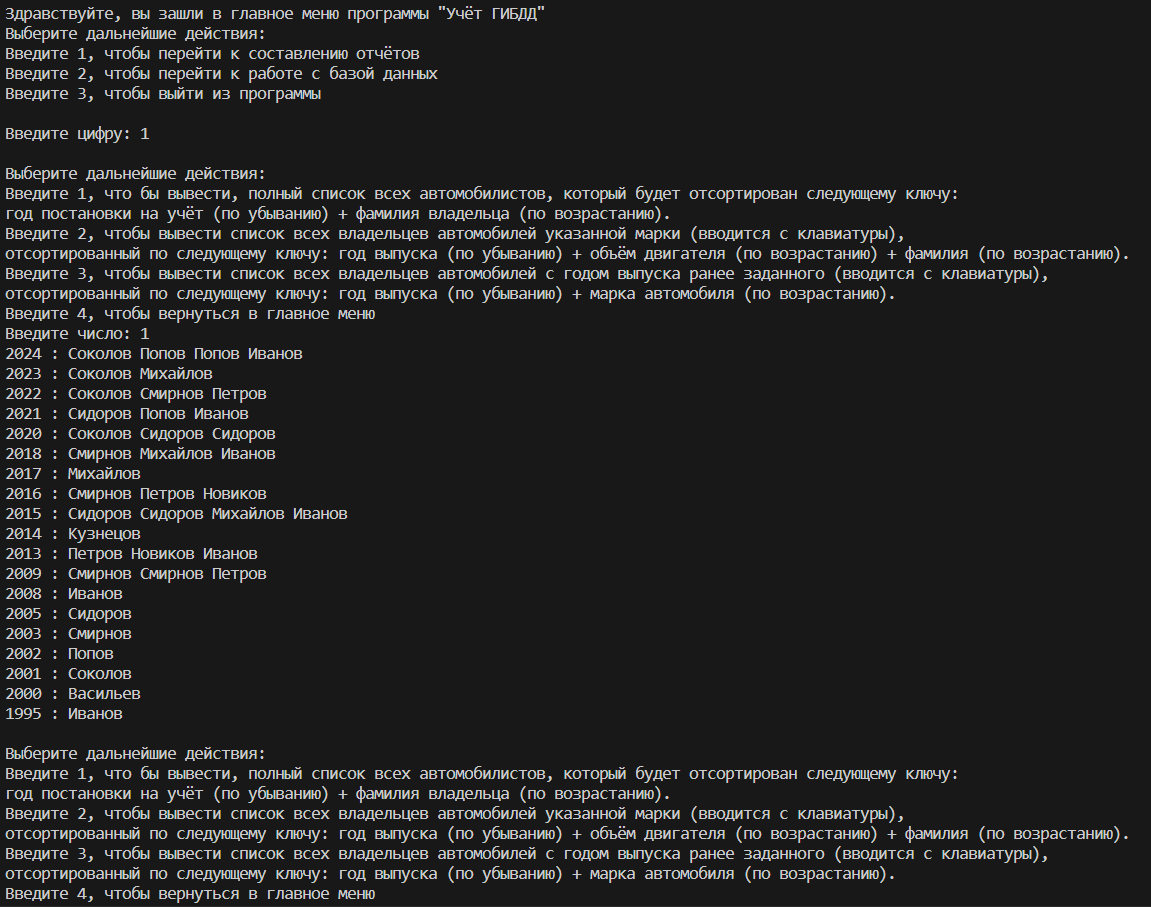
8. Создание функции database для выбора нужного нам действия при работе с данными файла.  
  
9. Создание функции add\_str\_database для добавления новой записи в файл с записями, содержащими информацию о водителях.

10. Создание функции del\_str\_database для удаления одной строки из файла с записями, содержащими информацию о водителях.

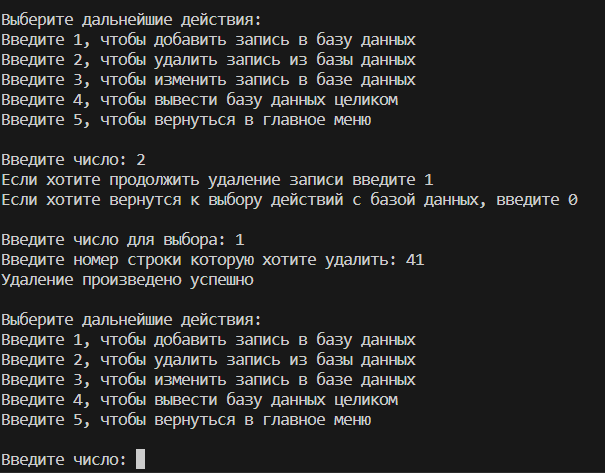
11. Создание функции change\_str\_database для изменения одной строки из файла с записями, содержащими информацию о водителях.

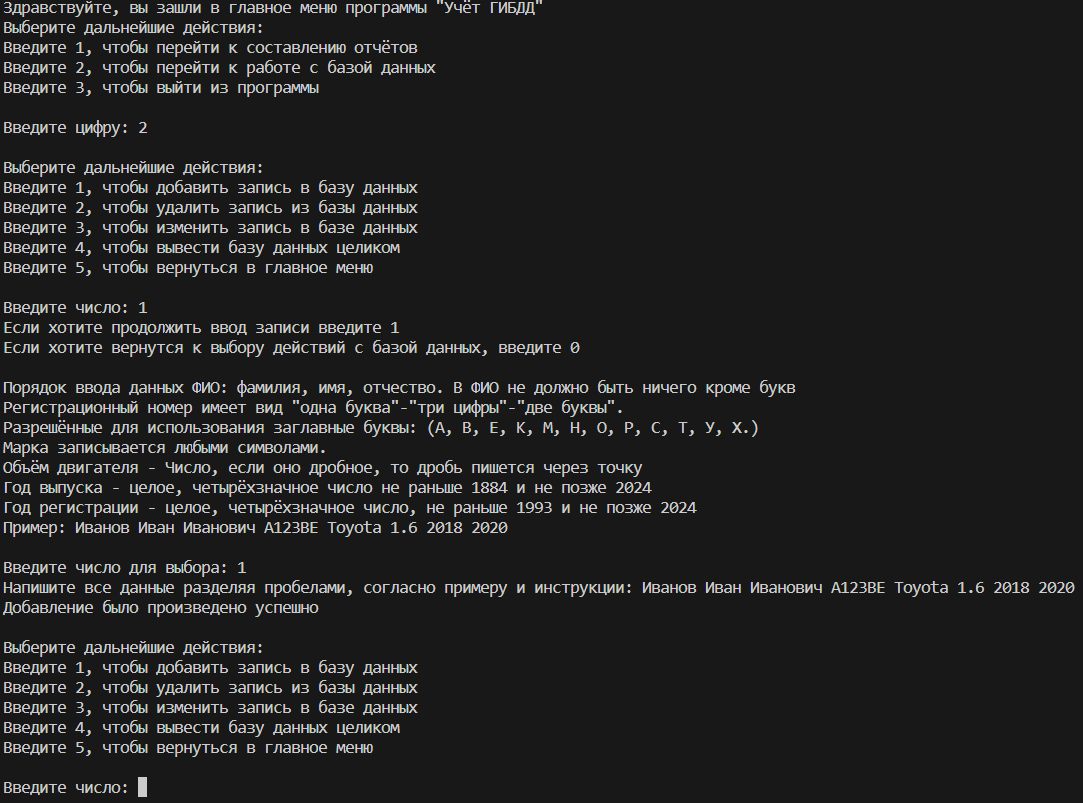
12. Создание функции check для проверки записей в файле. Функция используется в началае, при работе first\_check, которая проверяет корректность записей в файле перед работой с ними и при работе функций add\_str\_database и change\_str\_database.  
  
13. Тестирование программы, отладка работы, устранение неполадок

# Тестирование



# 





# Код программы

numbers = ['0','1','2','3','4','5','6','7','8','9', 0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

letters = ['A','А','B','В','E','Е','М','M','Н','H','О','O','P','Р','C','С','T','Т','У','Х','X',]

def first\_check(f\_check):

"""

Функция first\_check осуществляет первичную проверку данных. В случае, если f\_check равен 2, выводит сообщение о некорректности данных.

Если длина данных в файле 'data.txt' меньше 25 строк, также выводит сообщение об ошибке.

После проверки данных вызывает check для каждой строки данных.

После окончания проверки вызывает main\_menu.

"""

with open('data.txt', 'r', encoding="utf-8") as file:

data = file.readlines()

if len(data) < 25:

print('Строк в файле меньше, чем 25')

file.close()

exit

else:

for i in range(0, len(data)):

error = check(data[i], f\_check)

if error == 2:

break

if error == 2:

print('Данные в файле не корректные')

return None

else:

main\_menu(0)

def main\_menu(f\_check):

"""

Функция main\_menu представляет главное меню программы "Учёт ГИБДД".

Организует взаимодействие пользователя с программой, предлагая выбор дальнейших действий.

В зависимости от выбора выполняет соответствующую функцию.

"""

print('Здравствуйте, вы зашли в главное меню программы "Учёт ГИБДД"',

'\nВыберите дальнейшие действия:',

'\nВведите 1, чтобы перейти к составлению отчётов',

'\nВведите 2, чтобы перейти к работе с базой данных',

'\nВведите 3, чтобы выйти из программы',

'\n')

try:

choise\_n = int(input('Введите цифру: '))

except ValueError:

print('Вы ошиблись при вводе, попробуйте ещё раз')

main\_menu(f\_check)

else:

if choise\_n == 1:

reports(f\_check)

elif choise\_n == 2:

database(f\_check)

elif choise\_n == 3:

exit

else:

print('Вы ошиблись при вводе, попробуйте ещё раз')

main\_menu(f\_check)

def reports(f\_check):

"""

Функция reports предоставляет пользователю действия для работы с отчетами.

После выбора действия осуществляет вызов соответствующей функции.

"""

print('\nВыберите дальнейшие действия:',

'\nВведите 1, что бы вывести, полный список всех автомобилистов, который будет отсортирован следующему ключу:',

'\nгод постановки на учёт (по убыванию) + фамилия владельца (по возрастанию).',

'\nВведите 2, чтобы вывести список всех владельцев автомобилей указанной марки (вводится с клавиатуры),',

'\nотсортированный по следующему ключу: год выпуска (по убыванию) + объём двигателя (по возрастанию) + фамилия (по возрастанию).',

'\nВведите 3, чтобы вывести список всех владельцев автомобилей с годом выпуска ранее заданного (вводится с клавиатуры),',

'\nотсортированный по следующему ключу: год выпуска (по убыванию) + марка автомобиля (по возрастанию).',

'\nВведите 4, чтобы вернуться в главное меню')

try:

choise\_n = int(input('Введите число: '))

except ValueError:

print('Вы ошиблись при вводе, попробуйте ещё раз')

reports(f\_check)

else:

if choise\_n == 1:

report\_1(f\_check)

elif choise\_n == 2:

report\_2(f\_check)

elif choise\_n == 3:

report\_3(f\_check)

elif choise\_n == 4:

main\_menu(f\_check)

else:

print('Вы ошиблись при вводе, попробуйте ещё раз')

reports(f\_check)

def database(f\_check):

"""

Функция database позволяет пользователю осуществлять работу с базой данных.

В зависимости от выбора вызывает соответствующую функцию для добавления, удаления, изменения записей или просмотра базы данных целиком.

"""

print('\nВыберите дальнейшие действия:',

'\nВведите 1, чтобы добавить запись в базу данных',

'\nВведите 2, чтобы удалить запись из базы данных',

'\nВведите 3, чтобы изменить запись в базе данных',

'\nВведите 4, чтобы вывести базу данных целиком',

'\nВведите 5, чтобы вернуться в главное меню',

'\n')

try:

choise\_n = int(input('Введите число: '))

except ValueError:

print('Вы ошиблись при вводе, попробуйте ещё раз')

database(f\_check)

else:

if choise\_n == 1:

add\_str\_database(f\_check)

elif choise\_n == 2:

del\_str\_database(f\_check)

elif choise\_n == 3:

change\_str\_database(f\_check)

elif choise\_n == 4:

database\_view(f\_check)

elif choise\_n == 5:

main\_menu(f\_check)

else:

print('Вы ошиблись при вводе, попробуйте ещё раз')

database()

def report\_1(f\_check):

"""

Функция report\_1 формирует отчет о списках автомобилистов на основе определенных ключей и сортировки.

Результаты выводятся на экран.

"""

year\_reg\_dict = {}; year\_reg\_array = []; surname\_array = []; surname\_str = str(); last\_array = []

with open('data.txt', 'r+', encoding="utf-8") as file:

lines = file.readlines()

for i in range(0, len(lines)):

line = lines[i].split()

year\_reg\_array.append(int(line[7]))

year\_reg\_set = set(year\_reg\_array)

year\_reg\_array = list(year\_reg\_set)

for i in range(0,len(year\_reg\_array)):

for i1 in range(0, len(lines)):

line = lines[i1].split()

if str(year\_reg\_array[i]) == line[7]:

surname\_array.append(line[0])

surname\_array = sorted(surname\_array)

surname\_array.reverse()

for i1 in range(0,len(surname\_array)):

surname\_str += surname\_array[i1] + ' '

year\_reg\_dict[int(year\_reg\_array[i])] = surname\_str

surname\_str = str(); surname\_array = []

for i in range(0, len(year\_reg\_array)):

last\_str = str(str(int(year\_reg\_array[i])) + ' : ' + year\_reg\_dict[int(year\_reg\_array[i])])

last\_array.append(last\_str)

last\_array.sort()

last\_array.reverse()

for i in range(0,len(last\_array)):

print(last\_array[i])

file.close()

reports(f\_check)

def report\_2(f\_check):

"""

Функция report\_2 формирует отчет о владельцах автомобилей указанной марки с использованием заданных ключей для сортировки.

После сортировки результаты выводятся на экран.

"""

year\_rel\_array = []

volume\_array = []

surname\_array = []

try:

n\_brand = str(input('Введите название марки: '))

except ValueError:

print('Вы ввели некорректные данные, попробуйте ещё раз')

return

with open('data.txt', 'r', encoding="utf-8") as file:

for line in file:

line\_data = line.split()

if n\_brand == str(line\_data[4]):

year\_rel\_array.append(int(line\_data[6]))

volume\_array.append(float(line\_data[5]))

surname\_array.append(str(line\_data[0]))

n = len(year\_rel\_array)

if n == 0:

print('Машин такой марки нет в базе данных')

else:

for i in range(n):

for j in range(0, n-i-1):

if year\_rel\_array[j] < year\_rel\_array[j+1]:

year\_rel\_array[j], year\_rel\_array[j+1] = year\_rel\_array[j+1], year\_rel\_array[j]

volume\_array[j], volume\_array[j+1] = volume\_array[j+1], volume\_array[j]

surname\_array[j], surname\_array[j+1] = surname\_array[j+1], surname\_array[j]

for i in range(n):

print(f"Год выпуска: {year\_rel\_array[i]}, Объём двигателя: {volume\_array[i]}, Фамилия: {surname\_array[i]}")

file.close()

reports(f\_check)

def report\_3(f\_check):

"""

Функция report\_3 формирует отчет о владельцах автомобилей с заданным годом выпуска на основе ключей для сортировки.

После сортировки результаты выводятся на экран.

"""

year\_rel\_dict = {}; year\_brand\_array = []; year\_array = []; brand\_array = []; surname\_array = []; b\_n = brand\_str = str()

try:

year\_rel = str(input('Введите год, раньше которого вы хотите узнать: '))

except ValueError:

print('Вы ввели некорректные данные, попробуйте ещё раз')

report\_3(f\_check)

else:

if int(year\_rel) < 1884:

print('Слишком давно, попробуйте снова')

report\_3(f\_check)

else:

with open('data.txt', 'r+', encoding="utf-8") as file:

lines = file.readlines()

for i in range(0, len(lines)):

line = lines[i].split()

if int(year\_rel) > int(line[6]):

year\_array.append(int(line[6]))

surname\_array = [str(line[4]), str(line[0])]

year\_brand\_array.append([int(line[6]), surname\_array])

year\_array = set(year\_array)

year\_array = list(year\_array)

year\_array.reverse()

for i in range(0, len(year\_array)):

for i1 in range(0, len(year\_brand\_array)):

if year\_brand\_array[i1][0] == year\_array[i]:

brand\_array.append(year\_brand\_array[i1][1])

brand\_array = sorted(brand\_array)

for i1 in range(0, len(brand\_array)):

brand\_str += str(brand\_array[i1]) + ' '

year\_rel\_dict[int(year\_array[i])] = brand\_str

brand\_str = str(); brand\_array = []

for i in range(0,len(year\_array)):

print(str(str(year\_array[i]) + ' : ' + year\_rel\_dict[year\_array[i]]))

file.close()

reports(f\_check)

def database\_view(f\_check):

"Функция database\_view выводит всю базу данных из файла на экран"

with open('data.txt', 'r+', encoding="utf-8") as file:

lines = file.readlines()

for i in range (len(lines)):

print(lines[i])

database(f\_check)

def add\_str\_database(f\_check):

"""

Функция add\_str\_database позволяет добавить запись в базу данных.

После ввода данных производит их проверку и сохранение в файл.

"""

with open('data.txt', 'a', encoding="utf-8") as file:

print('Если хотите продолжить ввод записи введите 1',

'\nЕсли хотите вернутся к выбору действий с базой данных, введите 0',

'\n',

'\nПорядок ввода данных ФИО: фамилия, имя, отчество. В ФИО не должно быть ничего кроме букв',

'\nРегистрационный номер имеет вид "одна буква"-"три цифры"-"две буквы".',

'\nРазрешённые для использования заглавные буквы: (А, В, Е, К, М, Н, О, Р, С, Т, У, Х.)',

'\nМарка записывается любыми символами.',

'\nОбъём двигателя - Число, если оно дробное, то дробь пишется через точку',

'\nГод выпуска - целое, четырёхзначное число не раньше 1884 и не позже 2024',

'\nГод регистрации - целое, четырёхзначное число, не раньше 1993 и не позже 2024',

'\nПример: Иванов Иван Иванович А123ВЕ Toyota 1.6 2018 2020',

'\n ')

try:

choise\_n = int(input('Введите число для выбора: '))

except ValueError:

print('Вы ввели некорректные данные, попробуйте ещё раз')

add\_str\_database(f\_check)

else:

if choise\_n == 0:

database(f\_check)

elif choise\_n == 1:

add\_str = str(input('Напишите все данные разделяя пробелами, согласно примеру и инструкции: '))

add\_str\_array = add\_str.split()

if len(add\_str\_array) != 8:

print('Вы ввели некорректную запись, попробуйте ещё раз')

file.close()

add\_str\_database(f\_check)

else:

check(add\_str, f\_check)

add\_str = add\_str + '\n'

file.write(add\_str)

print('Добавление было произведено успешно')

else:

print('Вы ввели некорректные данные, попробуйте ещё раз')

add\_str\_database(f\_check)

file.close()

database(f\_check)

def del\_str\_database(f\_check):

"""

Функция del\_str\_database позволяет пользователю удалить запись из базы данных.

Осуществляет проверку вводимых данных и удаление выбранной записи.

"""

print('Если хотите продолжить удаление записи введите 1',

'\nЕсли хотите вернутся к выбору действий с базой данных, введите 0',

'\n')

try:

choise\_n = int(input('Введите число для выбора: '))

except ValueError:

print('Вы ввели некоректные данные, попробуйте ещё раз')

del\_str\_database(f\_check)

else:

if choise\_n == 0:

database(f\_check)

elif choise\_n == 1:

with open('data.txt', 'r', encoding="utf-8") as file:

try:

change\_str = int(input('Введите номер строки которую хотите удалить: '))

except ValueError:

print('Вы ввели некорректные данные, попробуйте ещё раз')

del\_str\_database(f\_check)

else:

files = file.readlines()

if len(files) == 25:

print('Удаление записи приведёт к наличию менее 25 строк в БД, поэтому это невозможно')

database(0)

elif change\_str < 1 and change\_str >(len(files)+2):

print('Такой строки нет в БД')

database(0)

else:

del\_el = files.pop(change\_str-1)

print('Удаление произведено успешно')

file.close()

with open('data.txt', 'w+', encoding="utf-8") as file\_sec:

for i in range(0,len(files)):

file\_sec.write(files[i])

file\_sec.close()

database(f\_check)

else:

print('Вы ввели некорректные данные, попробуйте снова')

del\_str\_database(f\_check)

def change\_str\_database(f\_check):

"""

Функция change\_str\_database позволяет пользователю изменить запись в базе данных.

Выполняет проверку вводимых данных и изменение выбранной записи, если условия выполнены.

"""

print('Если хотите продолжить изменение записи введите 1',

'\nЕсли хотите вернутся к выбору действий с базой данных, введите 0',

'\n')

try:

choise\_n = int(input('Введите цифру для выбора: '))

except ValueError:

print('Вы ввели некоректные данные, попробуйте ещё раз')

change\_str\_database(f\_check)

else:

if choise\_n == 0:

database(f\_check)

elif choise\_n == 1:

with open('data.txt', 'r', encoding="utf-8") as file:

files = file.readlines()

try:

change\_str\_n = int(input('Введите номер строки которую хотите изменить: '))

except ValueError:

print('Вы ввели некорректные данные, попробуйте ещё раз')

change\_str\_database(f\_check)

else:

if change\_str\_n < 1 and change\_str\_n >(len(files)+2):

print('Такой строки нет в БД')

database(0)

else:

print('Если хотите продолжить ввод записи введите 1',

'\nЕсли хотите вернутся к выбору действий с базой данных, введите 0',

'\nПорядок ввода данных ФИО: фамилия, имя, отчество. В ФИО не должно быть ничего кроме букв',

'\nРегистрационный номер имеет вид "одна буква"-"три цифры"-"две буквы".',

'\nРазрешённые для использования заглавные буквы: (А, В, Е, К, М, Н, О, Р, С, Т, У, Х.)',

'\nМарка записывается любыми символами.',

'\nОбъём двигателя - Число, если оно дробное, то дробь пишется через точку',

'\nГод выпуска - целое, четырёхзначное число не раньше 1884 и не позже 2024',

'\nГод регистрации - целое, четырёхзначное число, не раньше 1993 и не позже 2024',

'\nПример: Иванов Иван Иванович А123ВЕ Toyota 1.6 2018 2020',

'\n ')

try:

choise\_n = int(input('Введите цифры для выбора: '))

except ValueError:

print('Вы ввели некорректные данные, попробуйте ещё раз')

change\_str\_database(f\_check)

else:

if choise\_n == 0:

database(f\_check)

elif choise\_n == 1:

change\_str = str(input('Напишите все данные разделяя пробелами, согласно примеру и инструкции: '))

change\_str\_array = change\_str.split()

if len(change\_str\_array) != 8:

print('Вы ввели некорректную запись, попробуйте ещё раз')

file.close()

change\_str\_database(f\_check)

else:

check(change\_str, 3)

files[change\_str\_n-1] = change\_str + '\n'

print('Изменение произведено успешно')

change\_str\_database(f\_check

else:

print('Вы ввели некорректные данные, попробуйте ещё раз')

change\_str\_database(f\_check)

file.close()

with open('data.txt', 'w+', encoding="utf-8") as file\_sec:

for i in range(0,len(files)):

file\_sec.write(files[i])

file\_sec.close()

else:

print('Вы ввели некорректные данные, попробуйте снова')

change\_str\_database(f\_check)

def check(data\_str, f\_check):

"""

Функция check осуществляет проверку вводимых данных.

Проверяет валидность введенных данных, таких как ФИО, регистрационный номер, объем двигателя и даты.

При наличии ошибок ввода, возвращает сообщение об ошибке.

"""

data\_str = data\_str.split()

if len(data\_str) == 8:

try:

n\_surname = str(data\_str[0])

n\_name = str(data\_str[1])

n\_patronym = str(data\_str[2])

n\_reg = str(data\_str[3])

brand = str(data\_str[4])

n\_volume = float(data\_str[5])

year\_rel\_n = int(data\_str[6])

year\_reg\_n = int(str(data\_str[7]))

except ValueError:

if f\_check == 1:

return 2

elif f\_check == 0:

print('Вы ввели некорректные данные, попробуйте ещё раз')

add\_str\_database(f\_check)

else:

if n\_surname.isalpha() == False:

if f\_check == 1:

return 2

elif f\_check == 0:

print('В фамилии есть символы, которые не являются буквами, попробуйте ещё раз')

add\_str\_database(0)

elif f\_check == 3:

print('В фамилии есть символы, которые не являются буквами, попробуйте ещё раз')

change\_str\_database()

if n\_name.isalpha() == False:

if f\_check == 1:

return 2

elif f\_check == 0:

print('В имени есть символы, которые не являются буквами, попробуйте ещё раз')

add\_str\_database(0)

elif f\_check == 3:

print('В имени есть символы, которые не являются буквами, попробуйте ещё раз')

change\_str\_database()

if n\_patronym.isalpha() == False:

if f\_check == 1:

return 2

elif f\_check == 0:

print('В отчестве есть символы, которые не являются буквами, попробуйте ещё раз')

add\_str\_database(0)

elif f\_check == 3:

print('В отчестве есть символы, которые не являются буквами, попробуйте ещё раз')

change\_str\_database()

if not((n\_reg[0] in letters) and (n\_reg[1] in numbers) and (n\_reg[2] in numbers) and (n\_reg[3] in numbers) and (n\_reg[4] in letters) and (n\_reg[5] in letters)):

if f\_check == 1:

return 2

elif f\_check == 0:

print('Введённая строка не соответствует форме, попробуйте ещё раз')

add\_str\_database(0)

elif f\_check == 3:

print('Введённая строка не соответствует форме, попробуйте ещё раз')

change\_str\_database()

if year\_rel\_n < 1884 and year\_rel\_n > 2024:

if f\_check == 1:

return 2

elif f\_check == 0:

print('Число вне допустимого диапазона')

add\_str\_database(0)

elif f\_check == 3:

print('Число вне допустимого диапазона')

change\_str\_database()

elif year\_rel\_n < 1993 and year\_rel\_n > 2024:

if f\_check == 1:

return 2

elif f\_check == 0:

print('Число вне допустимого диапазона')

add\_str\_database(0)

elif f\_check == 3:

print('Число вне допустимого диапазона')

change\_str\_database()

else:

if f\_check == 1:

return 2

elif f\_check == 0:

print('Вы ввели некорректные данные, попробуйте ещё раз')

add\_str\_database(f\_check)

elif f\_check == 3:

print('Вы ввели некорректные данные, попробуйте ещё раз')

change\_str\_database(0)

first\_check(1)