Руководство разработчика к программе по управлению базой данных пользователей социальной сети

Разработчики:

Студенты БИВ171

Приходько Д.С.

Келлер В.А.

Кудрявцев Д.А.

МОСКВА 2018

Оглавление

[**1.** **Структура исходных данных** 3](#_Toc516775330)

[**2.** **Структура каталогов приложения** 4](#_Toc516775331)

[**3.** **Структура приложения** 4](#_Toc516775332)

[**4.** **Описание функций разработчика и фрагментов программы** 5](#_Toc516775333)

[**4.1** **Описание скрипта «database.py»** 5](#_Toc516775334)

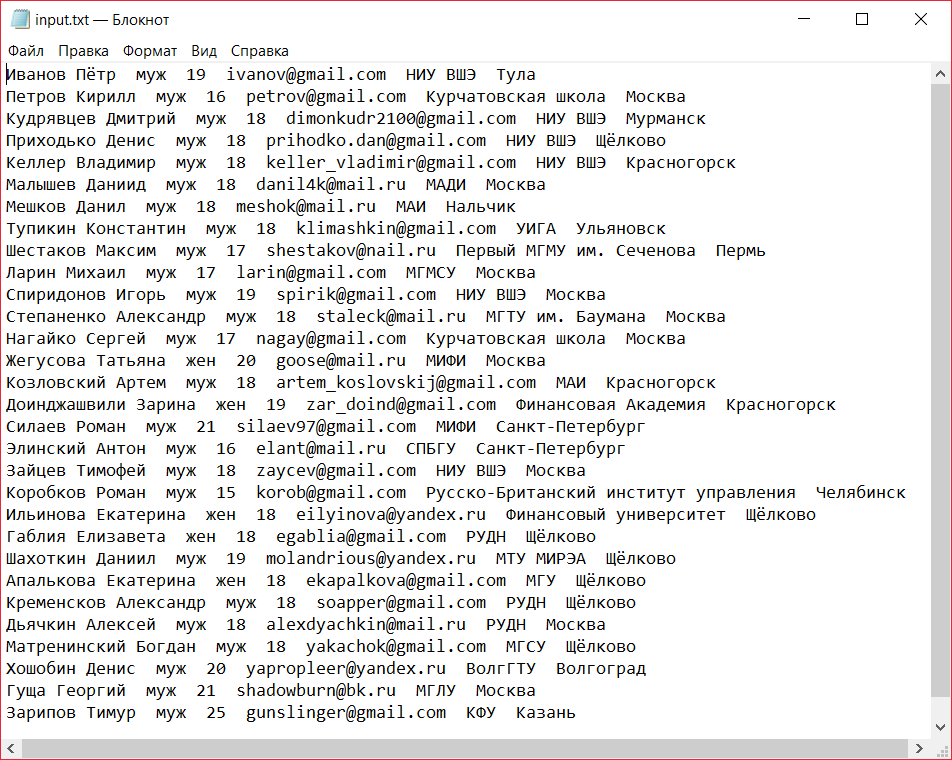
[**4.2** **Описание скрипта «Main.py»** 11](#_Toc516775335)

[**5.** **Пакеты и библиотеки, необходимые для работы приложения** 11](#_Toc516775336)

[**6.** **Технические требования** 12](#_Toc516775337)

# **Структура исходных данных**

Исходные данные, которые загружаются в программу при её запуске, должны содержаться в текстовом файле с названием «input.txt», расположенном в подкаталоге «Data». Структура файла должна выглядеть следующим образом:



В создаваемую базу данных заносятся следующие данные:

* «Имя» - фамилия и имя, разделённые одним знаком пробела
* «Пол» - пол пользователя соцсети (муж/жен)
* «Возраст» - числовое значение возраста пользователч соцсети
* «Почта» - полный e-mail адрес
* «Место учёбы» - название учебного заведения (допускается наличие одного пробела между словами)
* «Город» - город проживания

# **Структура каталогов приложения**

Все составные части программы хранятся в каталоге «Work», который может быть размещён в любой папке любого диска. Внутри этого каталога существует 5 подкаталогов:

* Data – здесь хранятся входные данные для приложения (текстовый файл с исходной базой), а также бинарный файл с зашифрованной в процессе работы программы базой данных
* Library – здесь хранится скрипт database.py, который используется как библиотека функций для управления базой данных
* Notes – папка, в которой находятся руководства пользователя и разработчика
* Output – в этой папке в процессе работы программы создаются текстовые файлы с изменённой базой данных и результатами поиска
* Scripts – каталог, внутри которого хранятся необходимые для запуска и работы программы скрипты

# **Структура приложения**

Приложение состоит из двух Python-скриптов. В подкаталоге «Library» находится скрипт, использующийся как стандартная библиотека функций. В этом файле описаны методы для работы с базой данных – зашифровка и расшифровка из бинарного файла, изменение, добавление и удаления полей базы данных, формирование текстовых файлов с результатами работы программы. Главный скрипт, «Main.py», содержащийся в подкаталоге «Scripts», содержит функции реализации графического интерфейса приложения, в которых происходит вызов функций взаимодействия с базой данных из скрипта «database.py».

# **Описание функций разработчика и фрагментов программы**

* 1. **Описание скрипта «database.py»**

Перед определением функций для работы с базой данных в скрипте находятся строки для подключения используемых стандартных библиотек языка Python:

**import** shelve **as** shl

**import** os

А также строка, в которой в переменную p сохраняется адрес текущего рабочего каталога (он понадобится в функциях для более удобной работы с файлами в различных подкаталогах).

p = os.getcwd()

Описание содержащихся в скрипте функций:

|  |  |  |  |
| --- | --- | --- | --- |
| Название функции | Входные данные | Возвращаемые данные | Описание |
| baseCreation | - | - | Создание бинарного файла, в котором содержится база данных |
| baseDecode | - | - | Загрузка файла базы данных в оперативную память из бинарного файла |
| baseEncode | b – база данных в виде словаря словарей | - | Зашифровка базы данных в бинарный файл |
| addPerson | b – база данных в виде словаря словарей, value – словарь с данными о новом пользователе | - | Добавление пользователя в базу данных |
| newBaseFile | b – база данных в виде словаря словарей | - | Запись в текстовые файлы (исходный и результат) изменённой базы данных |
| changePerson | b – база данных в виде словаря словарей, value – словарь с изменёнными данными о существующем пользователе | - | Изменение полей пользователя |
| deletePerson | b – база данных в виде словаря словарей, key – строка, содержащие фамилию пользователя, которого нужно удалить из базы данных | - | Удаление выбранного пользователя из базы данных |
| isPersonOk | per - словарь, содержащий одного пользователя, crit - строка, содержащая поле и значение критерия | True/False | Проверка удовлетворения пользователем критерия |
| siftedBase | fieldInput – строка, содержащая пользовательский ввод критерия | Новая база данных в виде словаря словарей | Поиск в базе данных по критериям введнным в поля ввода |
| statsCount | b – исходная база данных, new\_b – полученная в результате поиска база данных | - | Подсчёт итогов работы программы и занесение их в текстовый файл |

Затем, после описание функций, следуют строки, в которых база данных считывается из текстового файла, шифруется в бинарный, а затем заносится в оперативную память в виде словаря словарей:

baseCreation()

base = baseDecode()

Листинг приведённых в таблице функций:

**def** baseCreation():

    """

   Название: Создание бинарного файла, в котором содержится база данных

   Входные данные: -

   Возвращаемые данные: -

   Авторы: Кудрявцев Д.А., Приходько Д.С., Келлер В.А.

   """

**global** p

    fields = ["Имя", "Пол", "Возраст",  "Почта", "Место учёбы", "Город"]

    d = dict()

    input\_file = open(p + "**\\**Data**\\**input.txt", "r")

**for** name **in** input\_file:

        name = name.split("  ")

        name[2] = int(name[2])

        q = zip(fields, name)

        d[name[0].split()[0]] = dict(q)

    input\_file.close()

    f = shl.open(p + "**\\**Data**\\**base.shl", "n")

**for** x **in** d:

        f[x] = d[x]

    f.close()

**def** baseDecode():

    """

   Название: Загрузка файла базы данных в оперативную память

   Входные данные: -

   Возвращаемые данные: База данных в виде словаря словарей

   Автор: Приходько Д.С.

   """

**global** p

    f = shl.open(p + "**\\**Data**\\**base.shl")

    b = dict()

**for** x **in** f:

        b[x] = f[x]

**return** b

**def** baseEncode(b):

    """

   Название: Зашифровка базы данных в бинарный файл

   Входные данные: База данных в виде словаря словарей

   Возвращаемые данные: -

   Автор: Приходько Д.С.

   """

**global** p

    f = shl.open(p + "**\\**Data**\\**base.shl", "n")

**for** x **in** b:

        f[x] = b[x]

    f.close()

**def** addPerson(b, value):

    """

   Название: Добавление пользователя в базу данных

   Входные данные: База данных в виде списка списков

   Возвращаемые данные: Изменённая база данных

   Авторы: Келлер В.А., Приходько Д.С.

   """

    fields = ["Имя", "Пол", "Возраст",  "Почта", "Место учёбы", "Город"]

    b[value["Имя"].split()[0]] = dict()

    b[value["Имя"].split()[0]]["Имя"] = value["Имя"]

**for** x **in** fields[1:]:

**if** x == 2:

            b[value["Имя"].split()[0]][x] = int(value[x])

**else**:

            b[value["Имя"].split()[0]][x] = value[x]

    baseEncode(b)

    newBaseFile(b)

**def** newBaseFile(b):

    """

   Название: Запись в текстовые файлы (исходный и результат) изменённой базы данных

   Входные данные: База данных

   Возвращаемые данные: -

   Авторы: Приходько Д.С., Кудрявцев Д.А.

   """

**global** p

    f = open(p + "**\\**Output**\\**base.txt", "w")

    s = ""

**for** x **in** b:

**for** y **in** b[x]:

            s += str(b[x][y]) + "  "

        s += '**\n**'

    f.write(s)

    f.close()

    f = open(p + "**\\**Data**\\**input.txt", "w")

    s = ""

**for** x **in** b:

**for** y **in** b[x]:

            s += str(b[x][y]) + "  "

        s += '**\n**'

    f.write(s)

    f.close()

**def** changePerson(b, value):

    """

   Название: Изменение полей пользователя

   Входные данные: База данных как список списков, Фамилия и имя пользователя как его ключ в базе

   Возвращаемые данные: -

   Автор: Кудрявцев Д.А.

   """

    fields = ["Имя", "Пол", "Возраст", "Почта", "Место учёбы", "Город"]

**for** x **in** fields[1:]:

**if** x == 2:

            b[value["Имя"].split()[0]][x] = int(value[x])

**else**:

            b[value["Имя"].split()[0]][x] = value[x]

    baseEncode(b)

    newBaseFile(b)

**def** deletePerson(b, key):

    """

   Название: Удаление выбранного пользователя из базы данных

   Входные данные: База данных в виде словаря словарей

   Возвращаемые данные: Изменённый словарь

   Автор: Приходько Д.С.

   """

**del** b[key]

    baseEncode(b)

    newBaseFile(b)

**def** isPersonOK(per, crit, mode):

    """

   Название: Проверка удовлетворения пользователем критерия

   Входные данные: База данных в виде словаря словарей

   Возвращаемые данные: Истина или ложь в зависимости от удовлетворения условий

   Автор: Келлер В.А.

   """

    fields = ["Имя", "Пол", "Возраст", "Почта", "Место учёбы", "Город"]

**for** i **in** range(6):

**if** i == 2:

**if** len(crit[i]) != 0 **and** mode == 0 **and** (per[fields[i]] < int(crit[i][0]) **or** len(crit[i]) > 1):

**return** False

**elif** len(crit[i]) != 0 **and** mode == 1 **and** (per[fields[i]] > int(crit[i][0]) **or** len(crit[i]) > 1):

**return** False

**elif** len(crit[i]) != 0 **and** mode == 2 **and** (per[fields[i]] < int(crit[i][0]) **or** per[fields[i]] > int(crit[i][1]) **or** len(crit[i]) >2):

**return** False

**else**:

**if** len(crit[i]) != 0 **and** ' '.join(crit[i]) != per[fields[i]]:

**return** False

**return** True

**def** siftedBase(fieldInput, mode):

    """

   Название: Поиск в базе данных по критериям введнным в поля ввода

   Входные данные: Список полей ввода данных из графического интерфейса

   Возвращаемые данные: словарь словарей

   Автор: Келлер В.А.

   """

**try**:

        new\_b = dict()

        crit = []

**for** i **in** range(len(fieldInput)):

            crit.append(fieldInput[i].get().split())

**for** per **in** base.values():

**if** isPersonOK(per, crit, mode):

                new\_b[per["Имя"].split()[0]] = per

**return** new\_b

**except**:

**return** dict()

**def** statsCount(b, new\_b):

    """

   Название: Подсчёт итогов работы программы и занесение их в текстовый файл

   Входные данные: Исходная и полученная при поиске базы данных

   Возвращаемые данные: -

   Автор: Приходько Д.С., Кудрявцев Д.А., Келлер В.А.

   """

    s = 0

    avgAge = 0

    newsum = 0

    newfem = 0

    newmen = 0

    newAvgAge = 0

    newunis = dict()

    newcity = dict()

**for** i **in** b:

        s += 1

        avgAge += int(b[i]["Возраст"])

**if** s != 0:

        avgAge = round(avgAge / s)

**else**:

        avgAge = 0

**for** j **in** new\_b:

        newsum += 1

        newAvgAge += int(new\_b[j]["Возраст"])

        newunis[new\_b[j]["Место учёбы"]] = newunis.get(new\_b[j]["Место учёбы"], 0) + 1

        newcity[new\_b[j]["Город"]] = newcity.get(new\_b[j]["Город"], 0) + 1

**if** new\_b[j]["Пол"] == "жен":

            newfem += 1

**else**:

            newmen += 1

**print**(newunis)

**print**(newcity)

    max1 = max2 = 0

    city = ""

    univer = ""

**for** i **in** newunis:

**if** newunis.get(i,0) > max1:

            max1 = newunis.get(i, 0)

            univer = i

**for** i **in** newcity:

**if** newcity.get(i,0) > max2:

            max2 = newcity.get(i, 0)

            city = i

**print**(univer, city)

**if** newsum != 0:

        newAvgAge = round(newAvgAge / newsum)

**else**:

        newAvgAge = 0

    f = open(p + "**\\**Output**\\**result.txt", "w")

    st = ""

    st += "Общее количество записей в базе данных: " + str(s) + "**\n**"

    st += "Среднее значение возраста в базе данных: " + str(avgAge) + "**\n**"

    st += "Количество записей, найденных по заданным критериям: " + str(newsum) + "**\n**"

    st += "Среднее значение возраста найденных по заданным критериям: " + str(newAvgAge) + "**\n**"

    st += "Количество лиц мужского пола найденных по заданным критериям: " + str(newmen) + "**\n**"

    st += "Количество лиц женского пола найденных по заданным критериям: " + str(newfem) + "**\n**"

    st += "Большинство человек найденных по заданным критериям проживают в: " + city + "**\n**"

    st += "Большинство человек найденных по заданным критериям учаться в: " + univer + "**\n**"

    f.write(st)

    f.close()

* 1. **Описание скрипта «Main.py»**

В начале скрипта подключаются библиотеки tkinter для создания графического интерфейса и библиотека функций database, содержащая функции для работы с базой данных:

**from** tkinter **import** \*

**import** database **as** db

Импорт библиотек и добавление директории с database в PATH, чтобы появилась возможность импортировать её из другой папки:

**import** sys, os

os.chdir("..")

s = os.getcwd()

sys.path.append(s + "**\\**Library")

Описание содержащихся в скрипте функций:

|  |  |  |  |
| --- | --- | --- | --- |
| Название функции | Входные данные | Возвращаемые данные | Описание |
| delete | - | - | Удаляет из базы информацию о выбранном пользователе в списке пользователей |
| add | - | - | Открытия окна добавления.  Предлагает добавить информацию о новом человеке в базу данных |
| save | - | - | По нажатию кнопки “Сохранить” заносит информацию о пользователе в базу |
| change | - | - | Открытия окна изменения информации. Предлагает изменить информацию выделенного пользователя |
| set\_text | Текст для ввода, окно ввода | - | Записывает в поля ввода начальную информацию о выбранном пользвателе |
| search | - | - | Открытия окна поиска |
| process | - | - | Поиск и вывод найденной информации |
| on\_select2 | Действие | - | Выделение пользователя в списке найденным по критериям пользователей и вывод информации о нём в новом окне |
| on\_select | Действие | - | Выделение пользователя в списке пользователей базы данных и вывод информации о нём |

Затем, после описание функций, следуют строки, в которых создаётся главное окно для работы с базой данных:

**if** \_\_name\_\_ == "\_\_main\_\_":

    root = Tk()

    root.geometry("680x460")

    root.title("База данных")

    root.configure(bg='lightgrey')

    lbl1 = Label(root, text="Список людей:", fg='black', font=("Helvetica", 16, "bold"))

    lbl2 = Label(root, text="Информация:", fg='black', font=("Helvetica", 16, "bold"))

    lbl1.configure(bg='lightgrey')

    lbl2.configure(bg='lightgrey')

    lbl1.place(x=5, y=5)

    lbl2.place(x=200, y=5)

    scrollbar = Scrollbar(root, orient="vertical")

    listBox = Listbox(root, width=20, height=20, yscrollcommand=scrollbar.set, font=("Helvetica", 12))

    scrollbar.config(command=listBox.yview)

    scrollbar.pack(side="right", fill="y")

**for** names **in** db.base:

        listBox.insert(END, names)

    listBox.bind("<<ListboxSelect>>", on\_select)

    listBox.place(x=0, y=21)

    listSelection = Listbox(root, width=50, height=6, font=("Helvetica", 12))

    listBox.place(x=5, y=40)

    listSelection.place(x=200, y=40)

    addButton = Button(root, text="Добавить", command=add)

    addButton.place(x=20, y=430)

    searchButton = Button(root, text="Поиск", width=8, command=search)

    searchButton.place(x=90, y=430)

    exitButton = Button(root, text="Выход", command=quit)

    exitButton.pack(side="bottom", anchor="se")

    root.mainloop()

Листинг приведённых в таблице функций:

**def** delete():

    """

   Название: Удаление выбранного пользователя из базы данных

   Входные данные: -

   Возвращаемые данные: -

   Автор: Приходько Д.С., Кудрявцев Д.А.

   """

    listSelection.delete(0, END)

    selection = listBox.curselection()

**if** selection:

        value = listBox.get(selection)

        listBox.delete(selection[0])

        db.deletePerson(db.base, value)

**else**:

        listSelection.insert(END, "Выберите пользователя")

**def** add():

    """

   Название: Добавление пользователя в базу данных и в таблицу визуализации

   Входные данные: -

   Возвращаемые данные: -

   Автор: Келлер В.А., Приходько Д.С., Кудрявцев Д.А.

   """

    listSelection.delete(0, END)

    root = Tk()

    root.geometry("320x250")

    root.title("Добавить пользователя")

    root.configure(bg='lightgrey')

    field = ["Фамилия Имя", "Пол", "Возраст", "Почта", "Место учёбы", "Город"]

    x1 = 20

    y1 = 20

**for** i **in** field:

        lbl = Label(root, text=i, font=("Helvetica", 12))

        lbl.configure(bg='lightgrey')

        lbl.place(x=x1, y=y1)

        y1 += 30

    x1 = 140

    y1 = 20

    nameInput = Entry(root, width=20)

    nameInput.place(x=x1, y=y1)

    y1 += 30

    genderInput = Entry(root, width=20)

    genderInput.place(x=x1, y=y1)

    y1 += 30

    ageInput = Entry(root, width=20)

    ageInput.place(x=x1, y=y1)

    y1 += 30

    mailInput = Entry(root, width=20)

    mailInput.place(x=x1, y=y1)

    y1 += 30

    univerInput = Entry(root, width=20)

    univerInput.place(x=x1, y=y1)

    y1 += 30

    cityInput = Entry(root, width=20)

    cityInput.place(x=x1, y=y1)

    y1 += 30

**def** save():

        """

       Название: Сохранение данных после внесений информации в поля ввода по нажатию кнопки "Сохранить"

       Входные данные:

       Возвращаемые данные: -

       Автор: Кудрявцев Д.А., Приходько Д.С.

       """

**try**:

            value = dict()

            value["Имя"] = nameInput.get()

            value["Пол"] = genderInput.get()

            value["Возраст"] = int(ageInput.get())

            value["Почта"] = mailInput.get()

            value["Место учёбы"] = univerInput.get()

            value["Город"] = cityInput.get()

            listBox.insert(END, value["Имя"].split()[0])

            db.addPerson(db.base, value)

            root.destroy()

**except** ValueError:

**pass**

    saveButton = Button(root, text="Сохранить", command=save)

    saveButton.place(x=10, y=210)

    cancelButton = Button(root, text="Отмена", command=root.destroy)

    cancelButton.place(x=85, y=210)

    root.mainloop()

**def** change():

    """

   Название: Изменение полей данных пользователя по нажатию кнопки "Изменить" под его информацией

   Входные данные: -

   Возвращаемые данные: -

   Автор: Кудрявцев Д.А., Приходько Д.С.

   """

    selection = listBox.curselection()

**if** selection:

        value = listBox.get(selection)

        root = Tk()

        root.geometry("400x300")

        root.title("Изменить данные пользователя")

        root.configure(bg='lightgrey')

        field = ["Фамилия Имя", "Пол", "Возраст", "Почта", "Место учёбы", "Город"]

        x1 = 20

        y1 = 20

**for** i **in** field:

            lbl = Label(root, text=i, font=("Helvetica", 12))

            lbl.configure(bg='lightgrey')

            lbl.place(x=x1, y=y1)

            y1 += 30

        x1 = 140

        y1 = 20

**def** set\_text(text, e):

            """

           Название: Изначальная запись в поля ввода информации о пользователе

           Входные данные: Начальный текст поля для ввода, поле для ввода

           Возвращаемые данные: -

           Автор: Кудрявцев Д.А.

           """

            e.delete(0, END)

            e.insert(0, text)

        nameInput = Entry(root, width=20)

        nameInput.place(x=x1, y=y1)

        y1 += 30

        genderInput = Entry(root, width=20)

        genderInput.place(x=x1, y=y1)

        y1 += 30

        ageInput = Entry(root, width=20)

        ageInput.place(x=x1, y=y1)

        y1 += 30

        mailInput = Entry(root, width=20)

        mailInput.place(x=x1, y=y1)

        y1 += 30

        univerInput = Entry(root, width=20)

        univerInput.place(x=x1, y=y1)

        y1 += 30

        cityInput = Entry(root, width=20)

        cityInput.place(x=x1, y=y1)

        y1 += 30

        set\_text(db.base[value]["Имя"], nameInput)

        set\_text(db.base[value]["Пол"], genderInput)

        set\_text(db.base[value]["Возраст"], ageInput)

        set\_text(db.base[value]["Почта"], mailInput)

        set\_text(db.base[value]["Место учёбы"], univerInput)

        set\_text(db.base[value]["Город"], cityInput)

**def** save():

            """

           Название: Сохранение данных после изменений по нажатию кнопки "Сохранить"

           Входные данные: -

           Возвращаемые данные: -

           Автор: Кудрявцев Д.А., Приходько Д.С.

           """

**try**:

                info = dict()

                info["Имя"] = nameInput.get()

                info["Пол"] = genderInput.get()

                info["Возраст"] = int(ageInput.get())

                info["Почта"] = mailInput.get()

                info["Место учёбы"] = univerInput.get()

                info["Город"] = cityInput.get()

                db.changePerson(db.base, info)

                listSelection.delete(0, END)

                listSelection.insert(END, "Фамилия Имя: " + info["Имя"])

                listSelection.insert(END, "Пол: " + info["Пол"])

                listSelection.insert(END, "Возраст: " + str(info["Возраст"]))

                listSelection.insert(END, "Почта: " + info["Почта"])

                listSelection.insert(END, "Место учёбы: " + info["Место учёбы"])

                listSelection.insert(END, "Город: " + info["Город"])

                root.destroy()

**except** ValueError:

**pass**

        saveButton = Button(root, text="Сохранить", command=save)

        saveButton.place(x=10, y=270)

        cancelButton = Button(root, text="Отмена", command=root.destroy)

        cancelButton.place(x=85, y=270)

        root.mainloop()

**else**:

        listSelection.delete(0, END)

        listSelection.insert(END, "Выберите пользователя")

**def** search():

    """

   Название: Поиск и вывод информации по нажатию кнопки "Поиск"

   Входные данные: -

   Возвращаемые данные: -

   Автор: Кудрявцев Д.А., Келлер В.А., Приходько Д.С.

   """

    root = Tk()

    root.geometry("630x500")

    root.title("Окно поиска")

    root.configure(bg='lightgrey')

    lbl = Label(root, text="Введите критерий поиска:")

    lbl.place(x=0, y=0)

    lbl.configure(bg='lightgrey')

    lbl1 = Label(root, text="Поле данных")

    lbl1.place(x=230, y=10)

    lbl1.configure(bg='lightgrey')

**def** mode\_min():

**global** mode

        ds = Label(root, text="Текущий мод - min                ")

        ds.configure(bg='lightgrey')

        ds.place(x=380, y=120)

        mode = 0

**def** mode\_max():

**global** mode

        ds = Label(root, text="Текущий мод - max                ")

        ds.configure(bg='lightgrey')

        ds.place(x=380, y=120)

        mode = 1

**def** mode\_combo():

**global** mode

        ds = Label(root, text="Текущий мод - combobox")

        ds.configure(bg='lightgrey')

        ds.place(x=380, y=120)

        mode= 2

**def** mode\_normal():

**global** mode

        ds = Label(root, text="Текущий мод - normal         ")

        ds.configure(bg='lightgrey')

        ds.place(x=380, y=120)

        mode = -1

**def** process():

        """

       Название: Поиск и вывод информации по нажатию кнопки "Поиск"

       Входные данные: -

       Возвращаемые данные: -

       Автор: Кудрявцев Д.А., Келлер В.А.

       """

**def** on\_select2(event):

            """

           Название: Действие программы, когда пользователь выделяет поле в списке найденных пользователей

           Входные данные: Действие

           Возвращаемые данные: -

           Автор: Кудрявцев Д.А., Приходько Д.С.

           """

            w = event.widget

**try**:

                index = int(w.curselection()[0])

                value = w.get(index)

                listInfo.delete(0, END)

                listInfo.insert(END, "Фамилия Имя: " + db.base[value]["Имя"])

                listInfo.insert(END, "Пол: " + db.base[value]["Пол"])

                listInfo.insert(END, "Возраст: " + str(db.base[value]["Возраст"]))

                listInfo.insert(END, "Почта: " + db.base[value]["Почта"])

                listInfo.insert(END, "Место учёбы: " + db.base[value]["Место учёбы"])

                listInfo.insert(END, "Город: " + db.base[value]["Город"])

**except** IndexError:

**pass**

        new\_base = db.siftedBase(fieldInput, mode)

        db.statsCount(db.base, new\_base)

        lblres = Label(root, text="Результаты были занесены в текстовый файл result.txt", fg='black',

                       font=("Helvetica", 12, "bold"))

        lblres.place(x=0, y=250)

        lblres.configure(bg='lightgrey')

        lblw = Label(root, text="Результат поиска:", fg='black', font=("Helvetica", 12, "bold"))

        lblw.place(x=0, y=270)

        lblw.configure(bg='lightgrey')

        lblq = Label(root, text="Информация о пользователе:", fg='black', font=("Helvetica", 12, "bold"))

        lblq.place(x=200, y=270)

        lblq.configure(bg='lightgrey')

        scrollbar = Scrollbar(root, orient="vertical")

        listResult = Listbox(root, width=15, height=10, yscrollcommand=scrollbar.set, font=("Helvetica", 12))

        scrollbar.config(command=listResult.yview)

        scrollbar.pack(side="right", fill="y")

**for** names **in** new\_base:

            listResult.insert(END, names)

        listResult.bind("<<ListboxSelect>>", on\_select2)

        listResult.place(x=0, y=300)

        listInfo = Listbox(root, width=40, height=6, font=("Helvetica", 12))

        listInfo.place(x=200, y=300)

    field = ["Фамилия Имя", "Пол", "Возраст", "Почта", "Место учёбы", "Город"]

    fieldInput = []

    y1 = 30

    labels = []

    btns = []

**for** i **in** field:

        labels.append(Label(root, text=i))

        labels[-1].place(x=150, y=y1)

        labels[-1].configure(bg='lightgrey')

        fieldInput.append(Entry(root, width=20))

        fieldInput[-1].place(x=250, y=y1)

        y1 += 30

    minBtn = Button(root, text="min", command=mode\_min)

    minBtn.place(x=380, y=90)

    maxBtn = Button(root, text="max", command=mode\_max)

    maxBtn.place(x=415, y=90)

    comboBtn = Button(root, text="combobox", command=mode\_combo)

    comboBtn.place(x=450, y=90)

    normalBtn = Button(root, text="normal", command=mode\_normal)

    normalBtn.place(x=520, y=90)

**global** mode

    mode = -1

    btn = Button(root, text="Поиск", width=10, command=process)

    btn.place(x=225, y=210)

    root.mainloop()

**def** on\_select(event):

    """

   Название: Действие программы, когда пользователь выделяет поле в списке пользователей

   Входные данные: Действие

   Возвращаемые данные: -

   Автор: Кудрявцев Д.А., Приходько Д.С.

   """

    w = event.widget

**try**:

        index = int(w.curselection()[0])

        value = w.get(index)

        listSelection.delete(0, END)

        listSelection.insert(END, "Фамилия Имя: " + db.base[value]["Имя"])

        listSelection.insert(END, "Пол: " + db.base[value]["Пол"])

        listSelection.insert(END, "Возраст: " + str(db.base[value]["Возраст"]))

        listSelection.insert(END, "Почта: " + db.base[value]["Почта"])

        listSelection.insert(END, "Место учёбы: " + db.base[value]["Место учёбы"])

        listSelection.insert(END, "Город: " + db.base[value]["Город"])

**except** IndexError:

**pass**

    changeButton = Button(root, text="Изменить", command=change)

    changeButton.place(x=200, y=160)

    deleteButton = Button(text="Удалить", command=delete)

    deleteButton.place(x=270, y=160)

# **Пакеты и библиотеки, необходимые для работы приложения**

Программа использует встроенные функции языка Python, а также следующие его стандартные библиотеки:

* shelve – для шифрования базы данных в бинарный файл
* tkinter – для реализации графического интерфейса
* os – для возможности взаимодействия подкаталогов проекта между собой
* sys – для возможности импортирования своей библиотеки функций из другого подкаталога

# **Технические требования**

32- или 64-битная операционная система (Windows, Linux, MacOS и др.), на которую возможна установка программного обеспечения Python версии 3.5 или выше.