МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное Государственное Бюджетное Образовательное Учреждение Высшего Образования “Армавирский Государственный Педагогический Университет”

Кафедра Прикладной информатики, математики и физики

**КУРСОВАЯ РАБОТА**

**по дисциплине: «Базы данных»**

**на тему:**

**«Разработка базы данных информационной системы расчета стипендии»**

Выполнил:

студент 3 курса ВМ-ИВТ-2-1

Оганесян А.М.

Преподаватель:

директор института

Бельченко В.Е.

Армавир, 2020

**Оглавление:**

Глава 1. Введение и постановка задачи.

Глава 2. Анализ существующих систем.

Глава 3. Комплекс используемых программ и обоснование выбора среды проектирования.

Глава 4. Разработка программного приложения.

Глава 5. Выводы.

Глава 6. Список использованных источников.

Глава 7. Приложения.

**Глава 1. Введение и постановка задачи:**

Данная работа предлагает одно из решений проблемы создания собственной системы учета стипендий студентов.

Объектом исследования является создание базы данных и разработка системы управления ей с собственным интерфейсом и базой данных для учета стипендий студентов.

Предмет исследования – программа учета и анализа стипендий обучающихся в учебном заведении лиц.

Целью курсовой работы является разработка программного модуля, обеспечивающего учет стипендий студентов с возможностью удаленного администрирования.

Для раскрытия поставленной цели решается ряд конкретных задач, а именно:

Описание технологии создания и функционирования базы данных в сети интернет;

* Разработка функциональной модели системы;
* Разработка структуры базы данных;
* Разработка программного приложения.

Решение поставленных задач требует привлечения следующих методов: анализа соответствующей литературы по информатике, программированию, учебников и учебных пособий, обобщения и систематизации способов реализации. Возможности использования электронных справочников определяют практическую значимость курсовой работы.

При создании сервиса должны быть учтены следующие основные требования:

* оптимальное для пользователя представление исходной и конечной информации;
* простота и удобство в эксплуатации сервиса.

Поэтому, в качестве результатов курсовой работы выступают:

1. описание структуры программы управления базой данных;
2. разработка программных модулей;
3. возможность редактирования информации о стипендиях студента через сеть интернет.

**Глава 2. Анализ существующих систем:**

Почему не стоит использовать уже существующие системы?

1. Уникальность.

Когда вы скачиваете шаблон или покупаете его из общего доступа он, конечно же не удаляется. Поэтому вы вполне можете наткнуться на точно такую же систему управления бд в интернете. В контексте развития бизнеса и бренда это не несет вам преимуществ.

1. Брендинг.

В продолжение темы готовые решения не позволяют вам формировать собственный бренд в глазах аудитории. Может быть вам покажется это не таким важным фактором, но в долгосрочной перспективе это сильно влияет на успех.

1. Бизнес-ориентированность.

Кастомные решения хороши тем, что проводится бизнес-анализ для компании, выявляются потребности и проект затачивается конкретно для вас и ваших клиентов. Вам и вашим посетителям будет удобно и комфортно работать. Даже адаптивный шаблон (корректно отображается на мобильных устройствах) не позволит вам настроить все под себя, функциональность ограничена.

1. Интеграция.

Когда мы говорим о бизнесе, то кроме того вы работаете еще со многими инструментами: CRM, ERP-системы, социальные сети и другие вспомогательные сервисы. Очень полезно интегрировать их с сайтом, чего бесплатные шаблоны позволить себе не могут.

1. Юзабилити или же удобность использования.

Продолжая тему удобства для пользователей, разрабатывая по не по шаблону, специалисты могут правильно расположить CTA-элементы, различные блоки, которые будут привлекать клиентов и давать им нужную информацию. К тому же шаблоны не позволяют добавлять уникальные элементы, верстать текст, изображения, инфографики в современных стилях, которые будут интересны пользователям, как например разные виды слайдеров, анимации, динамики.

Всегда важно думать об удобстве клиента и о том, за сколько шагов он сможет достичь своих целей.

1. Конкуренция.

Если вы занимаетесь электронной коммерцией, то у вас может быть много конкурентов, которые предлагают аналогичные товары или услуги. Качество по играет важную роль, кастомизированность поможет вам удерживать посетителей и формировать их лояльность.

1. Тенденции.

Шаблоны довольно быстро устаревают и адаптировать их под новые тренды в дизайне или функциональности практически невозможно. Кастомизированное решение легко отредизайнить, добавить небольшую фичу, чтобы повысить интерес к ПО.

1. Юридические вопросы.

Покупая шаблон, будьте бдительны и проверьте следующую информацию, чтобы избежать проблем:

* кто обладает правами на шаблон
* можно ли вносить изменения
* нужно ли добавлять ссылки на авторов шаблона
* можно ли переводить шаблон на другой язык (важно, если вы хотите мультиязычное ПО).

Именно поэтому создание собственного ПО более перспективно нежели использование готовых шаблонов.

**Глава 3. Комплекс используемых программ и обоснование выбора среды проектирования:**

Для создания данного проекта использовался язык Python. Но почему именно он? За годы своего существования Python развился в динамичный, гибкий и очень мощный язык программирования. Многие разработчики предпочитают использовать для работы именно его, а не такие традиционные варианты, как C++ и Java. И среди разработчиков Python тоже очень популярен. Но поскольку сообщество не имеет единого мнения о том, насколько эффективно использовать Python для создания богатого функционалом ПО, давайте взвесим плюсы и минусы такого выбора.

Только сперва давайте обратим внимание на ведущие сайты и популярные веб-платформы, успешно использующие Python:

* **Instagram**. Эту невероятно популярную социальную сеть ежедневно посещают больше 4 млн. человек. Основная технология, которая использовалась при ее создании — Python.
* **Spotify**. Это приложение задает тон в индустрии стриминга музыки. Создано с использованием Python.
* **Disqus**. При помощи этого плагина пользователи ежемесячно оставляют 50 миллионов комментариев на сайтах. Основной язык программирования, использовавшийся при создании этого плагина, — Python.
* **YouTube**. Крупнейший видео хостинг YouTube является частью компании Google, которая во многих своих проектах использует язык Python и фреймворк Django.

Если посмотреть на приведенные примеры, становится ясно, что Python прекрасно подходит для создания самых разнообразных проектов. И совсем не удивительно, что он используется для различных направлений, включая путешествия, администрирование, образование, развлечения, здравоохранение, социальные медиа.

Но несмотря на многочисленные достоинства Python, многие разработчики все-таки для своих проектов выбирают другие языки программирования. Поэтому мы будем рассматривать не только плюсы, но и минусы использования Python.

Преимущества использования Python для разработки:

* **Простота использования**. Одна из самых привлекательных черт языка Python — невероятная простота использования в проектах. Дальше мы расскажем, с чем это связано.
* **Легкость изучения**. Этот язык программирования изучить не сложнее, чем самый обиходный английский. Простой синтаксис языка обеспечивает очень пологую кривую обучаемости.
* **Отлично подходит для визуализации данных**. Когда речь заходит о представлении данных на сайте или в приложении, Python просто потрясающе эффективен, и разработчики не могли этого не оценить. С его помощью можно легко создавать простые и понятные отчеты и визуальные представления данных.
* **Легкость чтения**. Простой синтаксис Python обеспечивает разработчикам отличную читаемость кода. Благодаря этому разработчикам, занятым в проекте, проще коммуницировать и обмениваться знаниями.
* **Модульность**. Для Python существует очень много различных модулей которые заметно упрощают работу в нём.
* **Несравненная гибкость**. Python это очень гибкий язык программирования. Он не замыкается в собственном мире и в процессе программирования способен смешиваться с разными языками, образуя новые реализации. Например, CPython — версия Python с C, IronPython создан для совместимости с .NET и C#, а проект PyObjc позволяет совместно использовать инструменты как Python, так и ObjectiveC. И это лишь несколько примеров.
* **Асинхронное программирование**. Все мы знаем, что шаблоны асинхронного программирования помогают решать многие проблемы, с которыми время от времени сталкиваются разработчики. Python поддерживает асинхронный код. А когда разные части программы могут работать отдельно друг от друга, это помогает выявлять и решать возникающие проблемы гораздо быстрее.

Минусы использования Python в разработке:

* **Ограничения скорости**. Поскольку Python это интерпретируемый язык программирования, программы на нем работают медленнее, чем написанные на компилируемых языках.
* **Проблемы с потоками**. Global Interpreter Lock (GIL) допускает выполнение только одного потока в каждый отдельный момент. Это создает заметные ограничения для использования языка Python.
* **Простота**. Хотя простота, на первый взгляд, это преимущество, она также является и недостатком. У программистов, привыкших к простому синтаксису, часто возникают трудности при переходе на более сложные языки, такие как Java.

По большому счету, преимущества языка Python перевешивают его недостатки. И он, благодаря своей непревзойденной гибкости, простоте использования и модульности, для многих разработчиков остается самым предпочтительным вариантом выбора в том числе и для меня.

Для более продуктивной работы я использовал в своей работе следующий модуль:

**Tkinter**. — кросс-платформенная событийно-ориентированная графическая библиотека на основе средств Tk (широко распространённая в мире GNU/Linux и других UNIX‐подобных систем, портирована также и на Microsoft Windows), написанная Стином Лумхольтом (Steen Lumholt) и Гвидо ван Россумом[1]. Входит в стандартную библиотеку Python. Tkinter плюсы:

1. Часть python, ничего лишнего для загрузки.
2. Очень простой синтаксис.
3. Текстовый виджет удивительно мощный и очень простой в работе.
4. Виджет canvas также очень прост и мощен. Ни один другой инструментарий не обеспечивает такого же сочетания простоты и мощности, как эти два виджета в моем опыте.
5. Использует собственные виджеты на mac и windows.
6. Tk - это твердая скала с несколькими кросс-платформенными особенностями. Я думаю, что механизм связывания Tkinter намного превосходит механизм wxPython; он более гибкий и последовательный. Я думаю, что три менеджера геометрии Tkinter-pack, place и grid-гораздо мощнее и проще в использовании, чем сайзеры wxPython.

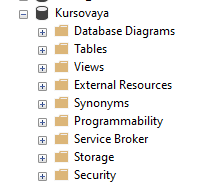
Так же в своей работе я использовал базу данных MSSQL на базе SQL SERVER. Почему я использовал именно SQL SERVER? Вот несколько преимуществ использования SQL SERVER:

1. SQL Server упрощает развертывание, передачу и интеграцию больших данных.
2. Интеграция структурированных и неструктурированных данных.
3. Высокая производительность.
4. Поддержка постоянной памяти (РМЕМ).
5. Гибридная транзакционная/аналитическая обработка (НТАР).
6. Интеллектуальная обработка запросов.
7. Безопасность и соответствие требованиям.
8. Azure Data Studio.
9. Выбор ОС и контейнеров.
10. Интеллектуальный анализ данных

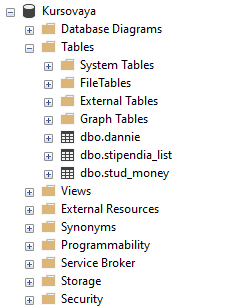
**Глава 4. Разработка программного обеспечения:**

Создание проекта началось с создания базы данных. Создание базы данных можно реализовать как через код, так и через визуальную оболочку MSSQL. Я же воспользовался вторым вариантом.

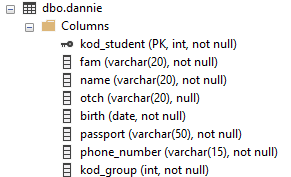
Подключаемся к нашему SQL серверу и в обозревателе объектов щелкаем по контейнеру «Базы данных» правой кнопкой мыши и выбираем пункт «Создать базу данных». Задаем имя нашей базе данных, в моем случае “Kursovaya”.



Далее мы создаем таблицы. В нашем случае мне понадобилось три таблицы: информация о студентах(таблица dannie), информация о существующих видах стипендий(таблица stipendia\_list) и таблица, в которой описываются связи студентов и стипендий:

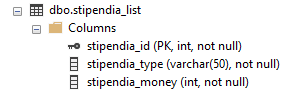


В таблице dannie мы создаем столбцы с информацией о студентах, в моем случае она такая:



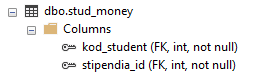
Так же мы даем каждому студенту индивидуальный код, по которому база данных будет определять уникальность студента.

В таблице stipendia\_list мы создаем 3 столбца с номером стипендии, её названием и суммой, которую получают студенты при ее наличии:



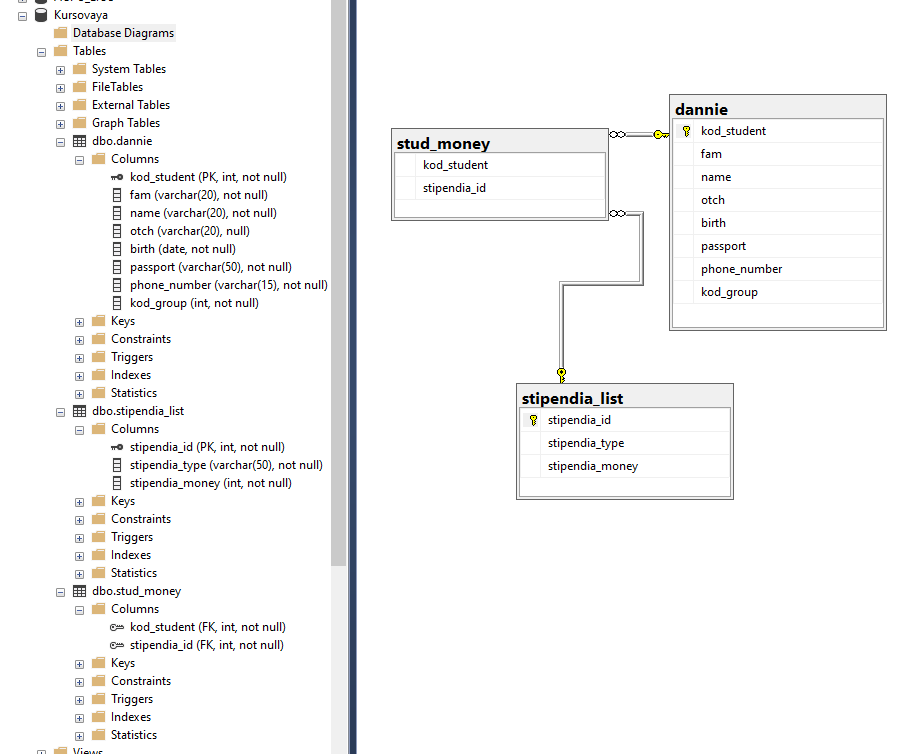
Так же, как и в данных о студентах, номер стипендии уникален.

В таблице stud\_money всего 2 столбца, и они позволяют связать уникальный код студента и стипендию:



Стоит обратить внимание, что эти столбцы напрямую зависят от столбцов таблиц к которым они привязаны и значения строк при изменении или удалении каскадно меняются или удаляются. При такой системе связей при удалении студента из базы, он удалится и из таблицы со стипендиями, что позволяет не выдавать стипендии несуществующим студентам или же выдавать несуществующие стипендии студентам.

Общий вид базы данных:



На этом этап разработки базы данных окончен и нужно написать ПО.

Весь код программы у меня уместился в один файл main.py.

Для начала нам нужно написать функцию подключения к нашей базе данных, и выглядит она так:

# Подключение к базе данных  
def create\_connection():  
 import pyodbc  
 server = 'sql-serverartem.ddns.net\ARTEM\_HOME\_SQL'  
 database = 'Kursovaya'  
 username = 'kursach'  
 password = 'kursach'  
 driver = '{SQL Server}'  
 port = '54432'  
 conn = pyodbc.connect(  
 'DRIVER=' + driver + ';SERVER=' + server + ';PORT=' + port + ';DATABASE=' + database + ';UID=' + username +  
 ';PWD=' + password)  
 return conn  
  
  
c = create\_connection()  
print('Подключение к базе данных успешно!')

При удачном подключении функция вернет нам курсор базы данных. Далее описываем интерфейс нашей программы:

root = Tk()  
  
f = Frame()  
f.pack(side=LEFT, padx=10)  
Label(f, text="Название стипендии").pack(fill=X)  
styp\_name = Entry(f)  
styp\_name.pack(anchor=N)  
Label(f, text="Сумма стипендии").pack(fill=X)  
styp\_money = Entry(f)  
styp\_money.pack(anchor=N)  
Button(f, text="Добавить новый тип стипендии", command=add\_item).pack(fill=X)  
Button(f, text="Удалить тип стипендии", command=del\_list).pack(fill=X)  
# Button(f, text="Save", command=save\_list) \  
# .pack(fill=X)  
box = Listbox(width=30, height=10)  
box.pack(side=LEFT)  
scroll = Scrollbar(command=box.yview)  
scroll.pack(side=LEFT, fill=Y)  
box.config(yscrollcommand=scroll.set)  
fs = Frame()  
fs.pack(side=LEFT, padx=10)  
Button(fs, text="Выбрать стипендию", command=choose\_stip).pack(fill=X)  
Button(fs, text="Выбрать студента", command=choose\_stud).pack(fill=X)  
Button(fs, text="Дать студенту стипендию", command=give\_stud\_stip).pack(fill=X)  
Button(fs, text="Забрать стиппендию у студента", command=del\_stud\_stip).pack(fill=X)  
Button(fs, text="Показать стипендии студента", command=check\_stud\_stip).pack(fill=X)  
  
boxst = Listbox(width=30, height=10)  
boxst.pack(side=LEFT)  
scrollst = Scrollbar(command=boxst.yview)  
scrollst.pack(side=LEFT, fill=Y)  
boxst.config(yscrollcommand=scrollst.set)  
  
fa = Frame()  
fa.pack(side=LEFT, padx=30)  
Label(fa, text="Имя").pack(fill=X)  
stud\_name = Entry(fa)  
stud\_name.pack()  
Label(fa, text="Фамилия").pack(fill=X)  
stud\_fam = Entry(fa)  
stud\_fam.pack()  
Label(fa, text="Отчество").pack(fill=X)  
stud\_otch = Entry(fa)  
stud\_otch.pack()  
Label(fa, text="День рождения").pack(fill=X)  
stud\_dr = Entry(fa)  
stud\_dr.pack()  
Label(fa, text="Паспортные данные").pack(fill=X)  
stud\_pd = Entry(fa)  
stud\_pd.pack()  
Label(fa, text="Номер телефона").pack(fill=X)  
stud\_num = Entry(fa)  
stud\_num.pack()  
Label(fa, text="Номер группы").pack(fill=X)  
stud\_gp = Entry(fa)  
stud\_gp.pack()  
Label(fa, text="Номер студенческого").pack(fill=X)  
stud\_sb = Entry(fa)  
stud\_sb.pack()  
fa = Frame()  
fa.pack(side=LEFT, padx=30)  
Button(fa, text="Добавить студента", command=add\_student).pack(fill=X)  
Button(fa, text="Удалить студента", command=del\_stud).pack(fill=X)  
stip\_list()  
stud\_list()  
root.mainloop()

Здесь указаны текстовые поля и кнопки для взаимодействия с базой данных. Но пока еще это все только пустой шаблон без функций и нашим кнопкам нужно добавить функциональности:

Кнопка добавления новой стипендии:



Код кнопки:

def add\_item():  
 if len(styp\_name.get()) > 4 and len(styp\_money.get()) > 0:  
 try:  
 maxz = c.execute("SELECT TOP 1 stipendia\_id FROM stipendia\_list ORDER BY stipendia\_id DESC").fetchone()[0]  
 maxz += 1  
 c.execute("INSERT INTO stipendia\_list VALUES (?,?,?)", (maxz, str(styp\_name.get()), str(styp\_money.get())))  
 c.commit()  
 box.insert(END, styp\_name.get())  
 styp\_name.delete(0, END)  
 styp\_money.delete(0, END)  
 except:  
 from tkinter import messagebox as mb  
 mb.showerror(title="ОШИБКА!",  
 message="ПРОВЕРЬТЕ ВВЕДЕНЫЕ ДАННЫЕ!\n"  
 "Автора данного проекта заставляют работать за бесплатно")

Кнопка удаления стипендии:



Код кнопки:

def del\_list():  
 global chose\_stip  
 select = list(box.curselection())  
 select.reverse()  
 if len(select) > 0:  
 for i in select:  
 sela = box.get(i)  
 c.execute("DELETE FROM stipendia\_list WHERE stipendia\_type = '" + str(sela) + "'")  
 c.commit()  
 box.delete(i)  
 chose\_stip = 'не выбрана'

Кнопка выбора стипендии:



Код кнопки:

def choose\_stip():  
 global chose\_stip  
 select = list(box.curselection())  
 select.reverse()  
 if len(select) > 0:  
 chose\_stip = box.get(select[0])  
 show\_info('Выбран: ' + str(chose\_stud) + ' и ' + str(chose\_stip) + ' стипендия')

Кнопка выбора студента:



Код кнопки:

def choose\_stud():  
 global chose\_stud, stud\_id  
 select = list(boxst.curselection())  
 select.reverse()  
 if len(select) > 0:  
 chose\_stud = boxst.get(select[0])  
 spl\_string = chose\_stud.split()  
 stud\_id = spl\_string[-1]  
 rm = spl\_string[:-1]  
 chose\_stud = ' '.join([str(elem) for elem in rm])  
 show\_info('Выбран: ' + str(chose\_stud) + ' и ' + str(chose\_stip) + ' стипендия')

Кнопка выдачи стипендии студенту:



Код кнопки:

def give\_stud\_stip():  
 if chose\_stip != 'не выбрана' and chose\_stud != 'никто':  
 stip\_id = c.execute("SELECT stipendia\_id FROM stipendia\_list WHERE stipendia\_type = '" + str(chose\_stip) + "'").fetchall()[0][0]  
 check = c.execute("SELECT kod\_student, stipendia\_id FROM stud\_money WHERE kod\_student = '" + str(stud\_id) + "' AND stipendia\_id = '" + str(stip\_id) + "'").fetchall()  
 if len(check) == 0:  
 c.execute("INSERT INTO stud\_money VALUES (?,?)", (str(stud\_id), str(stip\_id)))  
 c.commit()  
 show\_info('Стипендия успешно выдана данному студенту!')  
 else:  
 mb.showerror(title="ОШИБКА!",  
 message="Данный студент уже получает этот вид стипендии!")  
 else:  
 mb.showwarning(title="Осторожнее с кнопочками!",  
 message="Выберите студента или тип стипендии!")

Кнопка удаления стипендии у студента:



Код кнопки:

def del\_stud\_stip():  
 if chose\_stip != 'не выбрана' and chose\_stud != 'никто':  
 stip\_id = c.execute("SELECT stipendia\_id FROM stipendia\_list WHERE stipendia\_type = '" + str(chose\_stip) + "'").fetchall()[0][0]  
 check = c.execute("SELECT kod\_student, stipendia\_id FROM stud\_money WHERE kod\_student = '" + str(stud\_id) + "' AND stipendia\_id = '" + str(stip\_id) + "'").fetchall()  
 if len(check) != 0:  
 c.execute("DELETE FROM stud\_money WHERE kod\_student = '" + str(stud\_id) + "' AND stipendia\_id = '" + str(stip\_id) +"'")  
 c.commit()  
 show\_info('Данный студент больше не получает этот вид стипендии')  
 else:  
 mb.showerror(title="ОШИБКА!",  
 message="Данный студент не получает этот вид стипендии!")  
 else:  
 mb.showwarning(title="Осторожнее с кнопочками!",  
 message="Выберите студента или тип стипендии!")

Кнопка для проверки стипендий студента:



Код кнопки:

def check\_stud\_stip():  
 if stud\_id != 'никто':  
 check = c.execute("SELECT stipendia\_list.stipendia\_type, stipendia\_list.stipendia\_money "  
 "FROM stipendia\_list, stud\_money "  
 "WHERE stud\_money.stipendia\_id = stipendia\_list.stipendia\_id "  
 "AND stud\_money.kod\_student = '" + str(stud\_id) + "'").fetchall()  
 sum\_stip = 0  
 for i in check:  
 sum\_stip += i[1]  
 stip\_text = ''  
 for i in range(len(check)):  
 stip\_text += check[i][0]  
 if i != len(check) - 1:  
 stip\_text += ', '  
 show\_info('Стипендии студента: ' + str(stip\_text) + '\nна сумму: ' + str(sum\_stip) + ' рублей')  
 else:  
 mb.showwarning(title="Осторожнее с кнопочками!",  
 message="Выберите студента или тип стипендии!")

Кнопка удаления студента:



Код кнопки:

def del\_stud():  
 global chose\_stud  
 select = list(boxst.curselection())  
 select.reverse()  
 if len(select) > 0:  
 for i in select:  
 sela = boxst.get(i)  
 c.execute("DELETE FROM dannie WHERE kod\_student = '" + str(sela.split()[3]) + "'")  
 c.commit()  
 boxst.delete(i)  
 chose\_stud = 'никто'

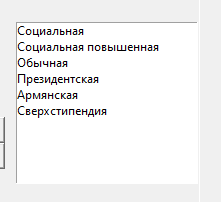
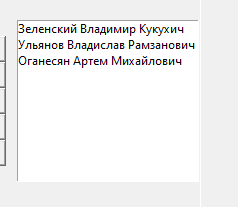
Кнопка добавления студента:



Код кнопки:

def add\_student():  
 if len(stud\_sb.get()) > 0:  
 try:  
 c.execute("INSERT INTO dannie VALUES (?,?,?,?,?,?,?,?)", (stud\_sb.get(), stud\_fam.get(), stud\_name.get(),  
 stud\_otch.get(), stud\_dr.get(), stud\_pd.get(),  
 stud\_num.get(), stud\_gp.get()))  
 c.commit()  
 boxst.insert(END, (stud\_fam.get() + ' ' + stud\_name.get() + ' ' + stud\_otch.get() + ' ' + str(stud\_sb.get())))  
 stud\_sb.delete(0, END)  
 stud\_fam.delete(0, END)  
 stud\_name.delete(0, END)  
 stud\_otch.delete(0, END)  
 stud\_dr.delete(0, END)  
 stud\_pd.delete(0, END)  
 stud\_num.delete(0, END)  
 stud\_gp.delete(0, END)  
 except:  
 mb.showerror(title="ОШИБКА!",  
 message="ПРОВЕРЬТЕ ВВЕДЕНЫЕ ДАННЫЕ!\n"  
 "Автора данного проекта заставляют работать за бесплатно")

Списки стипендий и студентов из базы данных:



Также функции выводящие списки стипендий и студентов из базы данных:

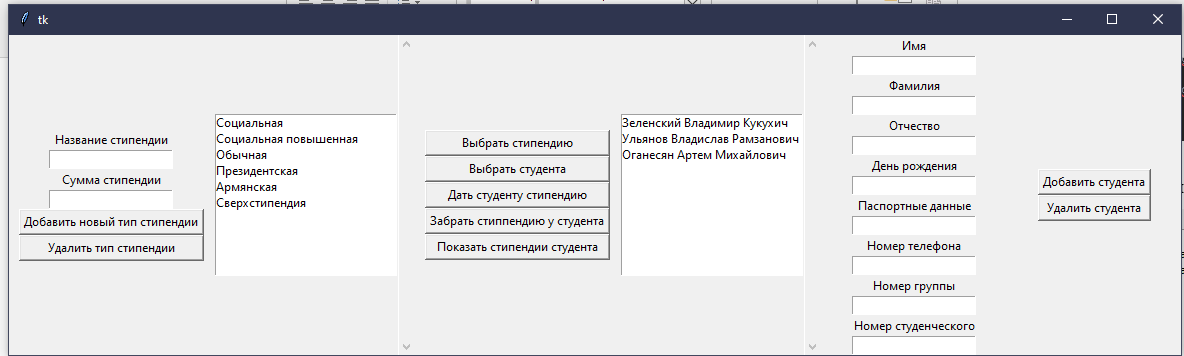
def stud\_list():  
 list\_stud = c.execute("SELECT fam, name, otch, kod\_student FROM dannie").fetchall()  
 for i in range(len(list\_stud)):  
 boxst.insert(END, (list\_stud[i][0] + ' ' + list\_stud[i][1] + ' ' + list\_stud[i][2] + ' ' + str(list\_stud[i][3])))  
  
  
def stip\_list():  
 list\_st = c.execute("SELECT stipendia\_type FROM stipendia\_list").fetchall()  
 for i in range(len(list\_st)):  
 box.insert(END, (list\_st[i][0]))

**Глава 5. Вывод:**

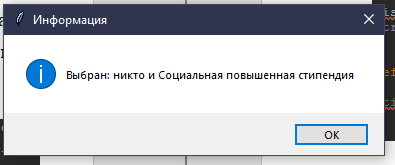
В итоге было разработано ПО с базой данных, в которой вы можете просмотреть виды стипендий, студентов и их стипендии.

Так же есть возможность удаленного администрирования для добавления или удаления стипендий или студентов.

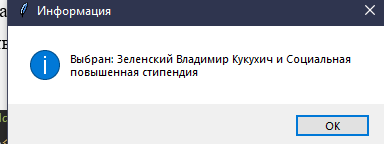
Финальный интерфейс программы выглядит так:



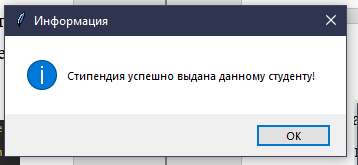
При выборе стипендии:



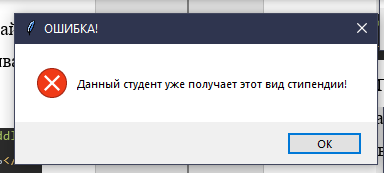
При выборе студента:



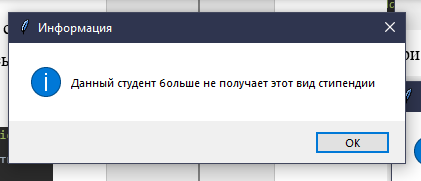
При даче студенту стипендии:



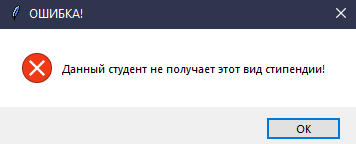
При даче стипендии, которая уже есть у студента:



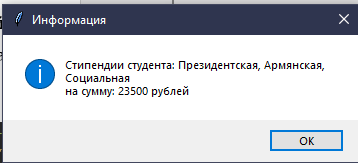
При отборе стипендии у студента:



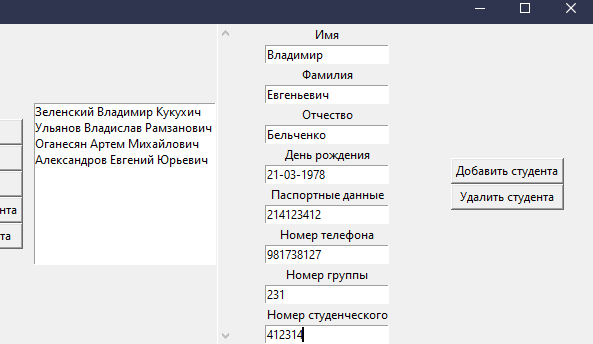
При попытке отобрать стипендию, которую не получает студент:



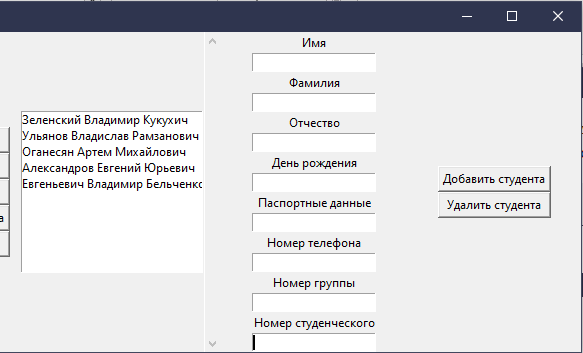
При запросе списка стипендий студента:



Поля добавления студента до нажатия на кнопку добавления:



Поля добавления студента после нажатия на кнопку добавления:



**Список литературных источников:**

*Сайты, порталы, базы данных*

eLIBRARY.RU: научная электронная библиотека: сайт. – Москва, 2000 – . – URL: <https://elibrary.ru> (дата обращения: 01.05.2020). – Режим доступа: для зарегистрир. пользователей. – Текст: электронный.

Министерство науки и высшего образования Российской Федерации: официальный сайт. – URL: [https://minobrnauki.gov.ru/](https://minobrnauki.gov.ru/%20%20)  (дата обращения: 01.05.2020). – Текст: электронный.

GitHub.com: крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки. URL: <https://github.com/krutoypan3> (дата обращения: 01.05.2020). – Текст: электронный.

*Компьютерные программы*

PyCharm: интегрированная среда разработки для языка программирования Python / разработчик “JetBrains”. – Электронная программа: электронная.

**Глава 7. Приложение**

Файл main.py:

from tkinter import \*  
import tkinter.messagebox as mb  
  
  
# Подключение к базе данных  
def create\_connection():  
 import pyodbc  
 server = 'sql-serverartem.ddns.net\ARTEM\_HOME\_SQL'  
 database = 'Kursovaya'  
 username = 'kursach'  
 password = 'kursach'  
 driver = '{SQL Server}'  
 port = '54432'  
 conn = pyodbc.connect(  
 'DRIVER=' + driver + ';SERVER=' + server + ';PORT=' + port + ';DATABASE=' + database + ';UID=' + username +  
 ';PWD=' + password)  
 return conn  
  
  
c = create\_connection()  
print('ok')  
  
  
def add\_item():  
 if len(styp\_name.get()) > 4 and len(styp\_money.get()) > 0:  
 try:  
 maxz = c.execute("SELECT TOP 1 stipendia\_id FROM stipendia\_list ORDER BY stipendia\_id DESC").fetchone()[0]  
 maxz += 1  
 c.execute("INSERT INTO stipendia\_list VALUES (?,?,?)", (maxz, str(styp\_name.get()), str(styp\_money.get())))  
 c.commit()  
 box.insert(END, styp\_name.get())  
 styp\_name.delete(0, END)  
 styp\_money.delete(0, END)  
 except:  
 from tkinter import messagebox as mb  
 mb.showerror(title="ОШИБКА!",  
 message="ПРОВЕРЬТЕ ВВЕДЕНЫЕ ДАННЫЕ!\n"  
 "Автора данного проекта заставляют работать за бесплатно")  
  
  
def del\_list():  
 global chose\_stip  
 select = list(box.curselection())  
 select.reverse()  
 if len(select) > 0:  
 for i in select:  
 sela = box.get(i)  
 c.execute("DELETE FROM stipendia\_list WHERE stipendia\_type = '" + str(sela) + "'")  
 c.commit()  
 box.delete(i)  
 chose\_stip = 'не выбрана'  
  
  
chose\_stip = 'не выбрана'  
chose\_stud = 'никто'  
stud\_id = ''  
  
  
def choose\_stip():  
 global chose\_stip  
 select = list(box.curselection())  
 select.reverse()  
 if len(select) > 0:  
 chose\_stip = box.get(select[0])  
 show\_info('Выбран: ' + str(chose\_stud) + ' и ' + str(chose\_stip) + ' стипендия')  
  
  
def show\_info(text):  
 msg = text  
 mb.showinfo("Информация", msg)  
  
  
def choose\_stud():  
 global chose\_stud, stud\_id  
 select = list(boxst.curselection())  
 select.reverse()  
 if len(select) > 0:  
 chose\_stud = boxst.get(select[0])  
 spl\_string = chose\_stud.split()  
 stud\_id = spl\_string[-1]  
 rm = spl\_string[:-1]  
 chose\_stud = ' '.join([str(elem) for elem in rm])  
 show\_info('Выбран: ' + str(chose\_stud) + ' и ' + str(chose\_stip) + ' стипендия')  
  
  
def give\_stud\_stip():  
 if chose\_stip != 'не выбрана' and chose\_stud != 'никто':  
 stip\_id = c.execute("SELECT stipendia\_id FROM stipendia\_list WHERE stipendia\_type = '" + str(chose\_stip) + "'").fetchall()[0][0]  
 check = c.execute("SELECT kod\_student, stipendia\_id FROM stud\_money WHERE kod\_student = '" + str(stud\_id) + "' AND stipendia\_id = '" + str(stip\_id) + "'").fetchall()  
 if len(check) == 0:  
 c.execute("INSERT INTO stud\_money VALUES (?,?)", (str(stud\_id), str(stip\_id)))  
 c.commit()  
 show\_info('Стипендия успешно выдана данному студенту!')  
 else:  
 mb.showerror(title="ОШИБКА!",  
 message="Данный студент уже получает этот вид стипендии!")  
 else:  
 mb.showwarning(title="Осторожнее с кнопочками!",  
 message="Выберите студента или тип стипендии!")  
  
  
def del\_stud\_stip():  
 if chose\_stip != 'не выбрана' and chose\_stud != 'никто':  
 stip\_id = c.execute("SELECT stipendia\_id FROM stipendia\_list WHERE stipendia\_type = '" + str(chose\_stip) + "'").fetchall()[0][0]  
 check = c.execute("SELECT kod\_student, stipendia\_id FROM stud\_money WHERE kod\_student = '" + str(stud\_id) + "' AND stipendia\_id = '" + str(stip\_id) + "'").fetchall()  
 if len(check) != 0:  
 c.execute("DELETE FROM stud\_money WHERE kod\_student = '" + str(stud\_id) + "' AND stipendia\_id = '" + str(stip\_id) +"'")  
 c.commit()  
 show\_info('Данный студент больше не получает этот вид стипендии')  
 else:  
 mb.showerror(title="ОШИБКА!",  
 message="Данный студент не получает этот вид стипендии!")  
 else:  
 mb.showwarning(title="Осторожнее с кнопочками!",  
 message="Выберите студента или тип стипендии!")  
  
  
def check\_stud\_stip():  
 if stud\_id != 'никто':  
 check = c.execute("SELECT stipendia\_list.stipendia\_type, stipendia\_list.stipendia\_money "  
 "FROM stipendia\_list, stud\_money "  
 "WHERE stud\_money.stipendia\_id = stipendia\_list.stipendia\_id "  
 "AND stud\_money.kod\_student = '" + str(stud\_id) + "'").fetchall()  
 sum\_stip = 0  
 for i in check:  
 sum\_stip += i[1]  
 stip\_text = ''  
 for i in range(len(check)):  
 stip\_text += check[i][0]  
 if i != len(check) - 1:  
 stip\_text += ', '  
 show\_info('Стипендии студента: ' + str(stip\_text) + '\nна сумму: ' + str(sum\_stip) + ' рублей')  
 else:  
 mb.showwarning(title="Осторожнее с кнопочками!",  
 message="Выберите студента или тип стипендии!")  
  
  
def del\_stud():  
 global chose\_stud  
 select = list(boxst.curselection())  
 select.reverse()  
 if len(select) > 0:  
 for i in select:  
 sela = boxst.get(i)  
 c.execute("DELETE FROM dannie WHERE kod\_student = '" + str(sela.split()[3]) + "'")  
 c.commit()  
 boxst.delete(i)  
 chose\_stud = 'никто'  
  
  
def add\_student():  
 if len(stud\_sb.get()) > 0:  
 try:  
 c.execute("INSERT INTO dannie VALUES (?,?,?,?,?,?,?,?)", (stud\_sb.get(), stud\_fam.get(), stud\_name.get(),  
 stud\_otch.get(), stud\_dr.get(), stud\_pd.get(),  
 stud\_num.get(), stud\_gp.get()))  
 c.commit()  
 boxst.insert(END, (stud\_fam.get() + ' ' + stud\_name.get() + ' ' + stud\_otch.get() + ' ' + str(stud\_sb.get())))  
 stud\_sb.delete(0, END)  
 stud\_fam.delete(0, END)  
 stud\_name.delete(0, END)  
 stud\_otch.delete(0, END)  
 stud\_dr.delete(0, END)  
 stud\_pd.delete(0, END)  
 stud\_num.delete(0, END)  
 stud\_gp.delete(0, END)  
 except:  
 mb.showerror(title="ОШИБКА!",  
 message="ПРОВЕРЬТЕ ВВЕДЕНЫЕ ДАННЫЕ!\n"  
 "Автора данного проекта заставляют работать за бесплатно")  
  
  
def stud\_list():  
 list\_stud = c.execute("SELECT fam, name, otch, kod\_student FROM dannie").fetchall()  
 for i in range(len(list\_stud)):  
 boxst.insert(END, (list\_stud[i][0] + ' ' + list\_stud[i][1] + ' ' + list\_stud[i][2] + ' ' + str(list\_stud[i][3])))  
  
  
def stip\_list():  
 list\_st = c.execute("SELECT stipendia\_type FROM stipendia\_list").fetchall()  
 for i in range(len(list\_st)):  
 box.insert(END, (list\_st[i][0]))  
  
  
root = Tk()  
  
f = Frame()  
f.pack(side=LEFT, padx=10)  
Label(f, text="Название стипендии").pack(fill=X)  
styp\_name = Entry(f)  
styp\_name.pack(anchor=N)  
Label(f, text="Сумма стипендии").pack(fill=X)  
styp\_money = Entry(f)  
styp\_money.pack(anchor=N)  
Button(f, text="Добавить новый тип стипендии", command=add\_item).pack(fill=X)  
Button(f, text="Удалить тип стипендии", command=del\_list).pack(fill=X)  
# Button(f, text="Save", command=save\_list) \  
# .pack(fill=X)  
box = Listbox(width=30, height=10)  
box.pack(side=LEFT)  
scroll = Scrollbar(command=box.yview)  
scroll.pack(side=LEFT, fill=Y)  
box.config(yscrollcommand=scroll.set)  
fs = Frame()  
fs.pack(side=LEFT, padx=10)  
Button(fs, text="Выбрать стипендию", command=choose\_stip).pack(fill=X)  
Button(fs, text="Выбрать студента", command=choose\_stud).pack(fill=X)  
Button(fs, text="Дать студенту стипендию", command=give\_stud\_stip).pack(fill=X)  
Button(fs, text="Забрать стиппендию у студента", command=del\_stud\_stip).pack(fill=X)  
Button(fs, text="Показать стипендии студента", command=check\_stud\_stip).pack(fill=X)  
  
boxst = Listbox(width=30, height=10)  
boxst.pack(side=LEFT)  
scrollst = Scrollbar(command=boxst.yview)  
scrollst.pack(side=LEFT, fill=Y)  
boxst.config(yscrollcommand=scrollst.set)  
  
fa = Frame()  
fa.pack(side=LEFT, padx=30)  
Label(fa, text="Имя").pack(fill=X)  
stud\_name = Entry(fa)  
stud\_name.pack()  
Label(fa, text="Фамилия").pack(fill=X)  
stud\_fam = Entry(fa)  
stud\_fam.pack()  
Label(fa, text="Отчество").pack(fill=X)  
stud\_otch = Entry(fa)  
stud\_otch.pack()  
Label(fa, text="День рождения").pack(fill=X)  
stud\_dr = Entry(fa)  
stud\_dr.pack()  
Label(fa, text="Паспортные данные").pack(fill=X)  
stud\_pd = Entry(fa)  
stud\_pd.pack()  
Label(fa, text="Номер телефона").pack(fill=X)  
stud\_num = Entry(fa)  
stud\_num.pack()  
Label(fa, text="Номер группы").pack(fill=X)  
stud\_gp = Entry(fa)  
stud\_gp.pack()  
Label(fa, text="Номер студенческого").pack(fill=X)  
stud\_sb = Entry(fa)  
stud\_sb.pack()  
fa = Frame()  
fa.pack(side=LEFT, padx=30)  
Button(fa, text="Добавить студента", command=add\_student).pack(fill=X)  
Button(fa, text="Удалить студента", command=del\_stud).pack(fill=X)  
stip\_list()  
stud\_list()  
root.mainloop()