Теоретическая  
часть  
проекта  
“теорема о:”

**Ученик:**Емельянцев М.Е.

**Преподаватель:**

Ахметзянов Ш.Ш.

Содержание:  
 Оглавление………………………………3

1.Основной алгоритм поиска………….5

2.GUI (Интерфейс)................................9

Основная суть и идея проекта  
  
 Центральная идея данного приложения -

дать простой и быстрый доступ к теоретическому материалу по стереометрии для учеников и учителей

при помощи языка Python

Задействованные библиотеки

Tkinter - кросс-платформенная событийно-ориентированная графическая библиотека на основе средств [Tk](https://ru.wikipedia.org/wiki/Tk) (широко распространённая в мире [GNU](https://ru.wikipedia.org/wiki/GNU)/[Linux](https://ru.wikipedia.org/wiki/Linux) и других [UNIX](https://ru.wikipedia.org/wiki/UNIX)‐подобных систем, портирована также и на [Microsoft Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows)), написанная Стином Лумхольтом (Steen Lumholt) и [Гвидо ван Россумом](https://ru.wikipedia.org/wiki/%D0%93%D0%B2%D0%B8%D0%B4%D0%BE_%D0%B2%D0%B0%D0%BD_%D0%A0%D0%BE%D1%81%D1%81%D1%83%D0%BC)[[1]](https://ru.wikipedia.org/wiki/Tkinter#cite_note-1). Входит в [стандартную библиотеку](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0) [Python](https://ru.wikipedia.org/wiki/Python).

Pandas - [программная библиотека](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0) на языке [Python](https://ru.wikipedia.org/wiki/Python) для обработки и [анализа данных](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Работа pandas с данными строится поверх библиотеки [NumPy](https://ru.wikipedia.org/wiki/NumPy), являющейся инструментом более низкого уровня. Предоставляет специальные структуры данных и операции для манипулирования числовыми [таблицами](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) и [временны́ми рядами](https://ru.wikipedia.org/wiki/%D0%92%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D0%B9_%D1%80%D1%8F%D0%B4). Название библиотеки происходит от [эконометрического](https://ru.wikipedia.org/wiki/%D0%AD%D0%BA%D0%BE%D0%BD%D0%BE%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D0%BA%D0%B0) термина [«панельные данные»](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D0%BD%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5), используемого для описания многомерных структурированных наборов информации. pandas распространяется под новой [лицензией BSD](https://ru.wikipedia.org/wiki/%D0%9B%D0%B8%D1%86%D0%B5%D0%BD%D0%B7%D0%B8%D1%8F_BSD).

PIL - библиотека языка [*Python*](https://ru.wikipedia.org/wiki/Python) (версии 2), предназначенная для работы с [растровой графикой](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0).

Разработка библиотеки прекращена (последняя правка датируется 2011 годом).[[1]](https://ru.wikipedia.org/wiki/Python_Imaging_Library#cite_note-hg-1) Однако проект под названием **Pillow**, являющийся [форком](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D0%BA) PIL, развивается и включает, в том числе, поддержку Python 3.x.[[2]](https://ru.wikipedia.org/wiki/Python_Imaging_Library#cite_note-pillow-2) Этот форк был принят в качестве замены оригинальной библиотеки и включён в некоторые дистрибутивы [Linux](https://ru.wikipedia.org/wiki/Linux), включая [Debian](https://ru.wikipedia.org/wiki/Debian_GNU/Linux)[[3]](https://ru.wikipedia.org/wiki/Python_Imaging_Library#cite_note-3) и [Ubuntu](https://ru.wikipedia.org/wiki/Ubuntu) (с [13.04](https://ru.wikipedia.org/w/index.php?title=Ubuntu_13.04_Raring_Ringtail&action=edit&redlink=1)).[[4]](https://ru.wikipedia.org/wiki/Python_Imaging_Library#cite_note-4)

Основной алгоритм поиска  
 1.Получение данных осуществляется при помощи библиотечной функции pandas

Данные включают в себя картинку, содержание теоремы, название и индекс  
class DB:

def \_\_init\_\_(self, filename):

self.df = pd.read\_excel(filename)

\*Примечание: Весь код написан при помощи ООП для достижения возможности дальнейшего улучшения, лучшей комплементации и упрощенного сопряжения разных библиотек   
 Далее описаны функции, которые по определенному признаку выдают варианты материала, содержащего введенные данные  
 2.Поиск по слову в названии  
def getTheoremByKeyWord(self, word):

results = []

for name, index in zip(self.df['Name'], range(len(self.df))):

if word.lower() in name.lower().split():

results.append(self.df.iloc[index]) return results

\*Примечание: В каждой из функций можно увидеть использование индекса, содержащегося в файле.Этот метод используется потому, что счетчик индекса совпадает с количеством элементов файла, из-за этого функция len нумерует элементы списка так, что их можно найти по внутре файловому индексу, привязанному к данным  
т.е.

1 название содержание картинка   
индекс 1 - строка 1 - номер элемента в цикле - 1  
и т.д.  
 3. Поиск названия по ключевому слову

def getNameByKeyWord(self, word):

results = []

for name, index in zip(self.df['Name'], range(len(self.df))):

if word.lower() in name.lower().split():

results.append(index)

results.append(name)

results.append('\n')

return results

\*Примечание: Данная функция нужна для заполнения выпадающего списка результатов  
В итоге будет сопряжена с предыдущей функцией, так как название легче передавать в отдельную функцию  
  
4.Поиск по подстроке

def getTheoremBySubStrInContext(self, substr):

results = []

for context, index in zip(self.df['Context'], range(len(self.df))):

if substr.lower() in context.lower():

results.append(self.df.iloc[index])

return results

\*Примечание: Данная функция предполагает, что введенная строка находиться в самой теореме т.е. её содержании. Её, функции, пока что нет в основном алгоритме, однако планируется её добавление в будущем

5. Поиск поиск по id  
 def getDataById(self, index):

return self.df.iloc[index]

GUI (Интерфейс)

Первой функцией, как и во всех классах ООП, является функция-конструктор \_\_init\_\_, которая задает объекты, с которыми работает класс. Также в оглавлении класса можно увидеть, что родительским классом является предыдущий, содержащий все функции поиска, что сделано для использования этих функций в этом классе  
class Quation(DB):

def \_\_init\_\_(self, main, fname):

super().init(fname) # Конструирование объекта базового класса

self.\_\_main = main

self.window\_width = 1400

self.window\_height = 900

self.\_\_main.geometry(str(self.window\_width) + 'x' + str(self.window\_height))

self.\_\_main.resizable(1, 1)

self.lst = []

self.labelphoto=Label()

self.labelphoto.grid(row=0,column=3,rowspan=10)

self.entry1 = Entry(main, font=15)

self.entry1.grid(row=0, column=0) self.textName = Text(main, font=15, wrap=WORD, height=2, width=50)

self.textName.grid(row=0, column=2)

self.textContent = Text(main, font=15, wrap=WORD, height=5, width=50)

self.textContent.grid(row=1, column=2, rowspan=5)

self.button1 = Button(main, text="Check")

self.button1.grid(row=0, column=1, rowspan=100) self.button1.config(command=self.show\_result)

\*Примечание:В этом классе объекты специфичны, так как использована библиотека Tkinter для формирования окна приложения.  
Тут можно увидеть основные настройки и изначальные кнопки. Последующие функции ссылаются исключительно на эти объекты при изменении и добавлении чего-либо в окне

1. Основная функция для кнопки показа результата

def show\_result(self):

txt = str(self.entry1.get())

res = self.getNameByKeyWord(word=txt)

for i in self.lst:

i.destroy()

self.lst = []

# print(len(self.lst))

# print(len(res),txt)

for i in range(len(res)):

if len(res[i])>50:

txt=res[i][:50]

else:

txt=res[i]

self.lst.append(Button(self.\_\_main, text=txt))

# print(res[i]['index'])

self.lst[i].config(command=lambda x=i: self.btn\_result\_clicked(res[x]))

self.lst[i].grid(row=i + 1, column=0)

\*Примечание:Данная функция создает список из кнопок, названия которых - результат действия функции

getNameByKeyWord

Далее кнопке присваивается функция self.btn\_result\_clicked

2.Функция нажатия кнопки из списка

def btn\_result\_clicked(self, index):

theorem\_name = self.getDataById(int(index))['Name'] # название теоремы

theorem\_content = self.getDataById(int(index))['Content'] # содержание теоремы

theorem\_img = self.getDataById(int(index))['Image'] # ссылка на картинку теоремы на жестком диске

self.textName.delete(1.0, END)

self.textName.insert(1.0, theorem\_name)

self.textContent.delete(1.0, END)

self.textContent.insert(1.0, theorem\_content)

self.photo=ImageTk.PhotoImage(Image.open(theorem\_img).resize((400,400)))

self.labelphoto.config(image=self.photo)

\*Примечание: Эта функция лишь заменяет содержание объектов из конструктора на содержание выбранной теоремы

3. Запуск программы

filename = 'theorems\_v2.xlsx'

root = Tk()

root.title('Теоремы')

q = Quation(root, filename)

root.mainloop()