## [ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ И НАУКИ ГОРОДА МОСКВЫ](https://www.mos.ru/donm/)

## Северо-Восточный административный округ

# Государственное бюджетное профессиональное образовательное учреждение города Москвы «Политехнический колледж имени П.А. Овчинникова»

**Специальность: 09.02.07 «Информационные системы и**

**программирование»**

**Курсовая работа**

**по «МДК 01.01»**

**Создание приложения с интерфейсом на языке программирования Python**

Выполнил: студент 2 курса

гр. 2ИС Фокин А.А.

Руководитель: Олейник П.В

МОСКВА 2022

**Оглавление**

1)Введение

2)Сущность языка программирования

3)Анализ существующих решений

4)Техническое требование

5)Проектирование программы

6) Разработка программы

7) Тестирование программы

8) Руководство по эксплуатации

9)Заключение

10)Список источников и литературы

**Введение**

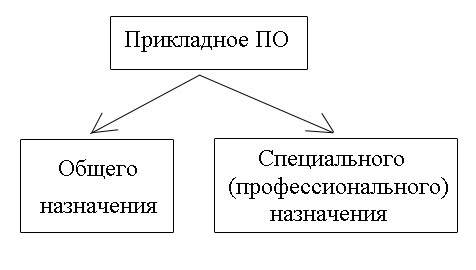
**Прикладное программное обеспечение**

Прикладное (специальное) программное обеспечение - это совокупность программ, разработанных при создании конкретной информационной системы.

Преднозначенно для решения конкретный задач пользователя и организации вычислительного процесса информационной системы в целом.

К прикладному программному обеспечению относятся компьютерные программы, написанные для пользователей или самими пользователями для задания компьютеру конкретной работы.

Различают следующую классификацию прикладного программного обеспечения:



**Прикладное ПО общего назначения-** совокупность программ для решения общих универсальных задач.Эти программы используются большинством пользователей компьютера.

**Виды прикладного ПО общего назначения:**

1.Текстовые редакторы

2.Текстовые процессоры

3.Электронные таблицы

4.Графические редакторы

5.Просмотрщики

6.Системное управление базами данных(СУБД)

7.Компьютерные игра

8.Переводчик

**Прикладное ПО специального (профессионального) назначения** - совокупность программ для решения более узких задач и профессиональных задач различных предметных областей (архитектуры, строительства, музыкальной и киноиндустрии).

**Виды прикладного ПО специального(профессионального) назначения:**

1.Настольные издательские системы

2.Электронные энциклопедии,учебники,словари,справочники

3.Система автоматизированного перевода(САТ-программы)

4.Серверные СУБД (клиент-серверные)

5.Редакторы трехмерной (3D) графики

6.Видеоредакторы(системы видеомонтажа)

7.Аудиоредакторы(системы аудиомонтажа)

8.Нотные редакторы

9.Экспертные системы

10.Системы автоматизированного проектирования(САПР,CAD\CAM\CAE)

**Целью данной** курсовой работы является разработка программы с графическим интерфейсом на языке программирования высокого уровня Python.

**Задачами данной курсовой работы являются:**

- выделение необходимых библиотек для решения заданной задачи;

- построение алгоритма программы для его реализации;

- программная реализация задачи при помощи Python.

**Целью исследования является:** разработка и реализация на языке высокого уровня алгоритма решения задачи, представленной в задании курсовой работы.

**Объектами исследования** настоящей курсовой работы являются

методы и технологии разработки программных продуктов.

**Предметами исследования** настоящей курсовой работы являются:

методы, алгоритмы и приёмы разработки программ графическим интерфейсом.

**Моя задача это создать красивый интерфейс и функциональный приложения состоящего из множества алгоритмов.**

## 

## 

## Сущность языка программирования

Язык программирования является формальным языком, предназначенным для записи компьютерных программ, он определяет набор лексических, синтаксических и семантических правил, определяющих внешний вид программы и действия, которые будут выполняться исполнителем.

Большая часть языков программирования состоят из инструкций для компьютера. Существуют программируемые машины, которые используют специфические инструкции, а не языки программирования.

Описание программного языка как правило разделяется на два компонента синтаксис (форма) и семантика (значение). Некоторые языки четко определены при помощи документаций, например, ISO стандарты в случае с языком программирования C, тогда как другие используют декларативные формы, то есть описывается, что представляет собой проблема и ожидаемый результат.

Языки программирования обладают следующими характерными чертами:

Функции и цели – язык компьютерного программирования используются для взаимодействия между человеком и машиной, в большинстве случаев в роли машины выступает компьютер.

Абстракция – языки программирования обычно содержат абстракции для обозначения и управления структурой данных и порядком управления.

Сила выражений – теория алгоритмов классифицирует языки программирования, по алгоритмам, которые они способны выражать.

История языков программирования начинается с 1940 года. В этом году был создан первый современный компьютер, но мощности компьютера на тот момент не хватало для выполнения программ на языке ассемблера и было необходимо вручную переписывать команды на язык нулей и единиц.

Конрад Цузе создал свой высокоуровневый язык Планкалкюль (Plankalkül) в 1943-1945 для своего Z1 компьютера, но язык не был имплементирован.

Первые функционирующие языки программирования появились в 1950-х годах. Short Code Джона Уильям Мокли, предложенный в 1949 году, был одним из первых высокоуровневых языков для электронного компьютера. В отличие от машинного кода, выражения в Short Code представляли из себя понятные математические выражения. Однако при каждом запуске программы она переводилась в машинный код что существенно замедляло скорость работы и по сравнению с эквивалентным машинным кодом Short Code был медленнее.

В ранних 1950-х, Алик Глени разработал Autocode, возможно первый компилируемый язык программирования. В 1954 году вторая итерация данного языка известная как Mark 1 Autocode была разработана для Mark 1 Брукером Р.А. Затем в 1950-х вместе с Университетом Манчестера Брукер разработал Autocode для Ферранти Меркури. Версия для EDSAC 2 была разработана Дугласом Харти из Кембриджского Университета.

В 1954 году командой под лидерством Джона Бакус из IBM был изобретен FORTRAN. Это был первый повсеместно используемый высокоуровневый язык программирования с имплементацией функций. Данный язык все еще используется для написания программ оценки производительности суперкомпьютеров.

Период с 1960 по 1970 годов принес расцвет языков программирования. Большинство современных парадигм программирования появились именно в этот период.

Speakeasy – разработан в 1964 году в ANL Стэнли Кохен, является объектно-ориентированным языком программирования. Использует синтаксис Fortran как свою основу. Данный язык всё еще используется.

Simula – разработан в поздних 1960-х Нигардом и Далем как надстройка Algol 60, был первым языком с поддержкой объектно-ориентированного программирования.

C – компилируемый статически типизированный язык программирования общего назначения, разработанный в 1969—1973 годах сотрудником Bell Labs Деннисом Ритчи как развитие языка B.

Smalltalk – середина 1970-х, предоставил фундаментальный дизайн объектно-ориентированного языка.

Prolog – разработан в 1972 году Колмерауером, Рузелом и Ковальски, является первым логическим языком программирования.

ML – разработан Робином Милнером в 1973 году, разработал статическую типизацию функциональных языков программирования.

Большинство современных языков являются родственниками хотя бы одного из перечисленных языков.

В 1960-1970-х также проводились дебаты о преимуществах «структурированного программирования», что означает программирование без использования «goto». Большая часть программистов верила в то, что использование «goto» является плохим стилем программирования за исключением некоторых случаев. Причина возникновения подобных дебатов являлось то, что некоторые языки программирования не использовали «goto», что заставляло программистов использовать структурированный подход к программированию.

В 1980-ые вместо создания новых парадигм, все начали более тщательно прорабатывать идеи разработанные в прошлом десятилетие. C++ совместило объектно-ориентированное программирование и системное программирование. США стандартизировали Ada. Многие страны начали инвестировать в пятое поколение языков программирования.

Одним из важнейших новых трендов является дизайн языков с фокусом на большие системы через использование модулей или больших организационных юнитов кода. Ada и ML использовали модульные системы.

Пусть в этот период и не появилось новых крупных парадигм, но разработчики расширили идеи старых парадигм и адаптировали их под современные условия.

В 1990-х благодаря росту популярности сети интернет, открылись новые платформы для компьютерных систем. Интернет дал возможность новым языкам таким как JavaScript получить популярность благодаря ранней интеграции с Netscape Navigation браузером. Различные другие языки получили распространение в разработки специальных приложений для веб-серверов, например, PHP.

Более радикальными и инновационными чем RAD языки стали новые сценарные языки. Они имели новый синтаксис и более свободное объединение функций. Сценарные языки стали самыми выдающимися языками для веб разработки.

В настоящее время присутствуют определенные тренды развития языков программирования, такие как:

* увеличение поддержки функционального программирования в коммерческих популярных языках, включая чистоту функционального программирования чтобы сделать код легче для восприятия и распараллеливать на микро- и макро- уровнях;
* конструкции для поддержки параллельного и распределенного вычисления;
* механизмы для улучшения безопасности и надежности верификаций языка: увеличенная проверка статичных данных, зависимого типа, контроль информационного потока, статичная потоковая безопасность;
* альтернативные механизмы для компонуемой и модульности: примесь, типаж, классы типов, делегации, аспекты;
* компонент ориентированная разработка ПО;
* метапрограммирование, рефлексия или доступ к абстрактному синтаксическому дереву;
* увеличение интереса в распространение и мобильности;
* интеграция с базами данных;
* открытое программное обеспечение как философия языков программирования;
* исследование квантовых компьютеров;
* повышенный интерес в визуальных языках программирования.

Каждый из языков программирования следует хотя бы одной из парадигм программирования. Парадигма программирования – это совокупность идей и понятий, определяющих стиль написания компьютерных программ. Выделяют следующие основные парадигмы:

• императивная парадигма – исходный код состоит из команд, которые выполняются последовательно, а результат этих команд может быть записан в память и читаться последующими инструкциями;

• декларативная парадигма – в данном стиле написания исходного кода задаётся спецификация решения задачи, то есть описывается, что представляет собой проблема и ожидаемый результат;

• структурная парадигма – в основе лежит представление программы в виде иерархической структуры блоков;

• функциональная парадигма – противопоставляется императивной парадигме;

• логическая парадигма – основа на автоматическом доказательстве теорем;

• объектно-ориентированная парадигма - основана на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования.

Практически все современные языки программирования в той или иной степень допускают использование различных парадигм, также парадигмы зачастую пересекаются в деталях. По статистике tiobe.com из 5 самых популярных языков программирования 4 являются объектно-ориентированными языками.

Специфика объектно-ориентированной парадигмы программирования

Объектно-ориентированное программирование является одной из методологий (парадигм) разработки программного обеспечения, в эту методологию входят языки, основанные на концепции «объектов» – условных сущностей, которые объединяют в себе поля (данные) и методы (выполняемые объектом действия).

Достаточно сложно конкретно определить, что такое объектно-ориентированный язык программирования, одним из основных подходов являются 6 принципов Алана Кэя, отца основателя ООП.

1. Всё является объектом.
2. Вычисления осуществляются путём взаимодействия (обмена данными) между объектами, при котором один объект требует, чтобы другой объект выполнил некоторое действие. Объекты взаимодействуют, посылая и получая сообщения. Сообщение — это запрос на выполнение действия, дополненный набором аргументов, которые могут понадобиться при выполнении действия.
3. Каждый объект имеет независимую память, которая состоит из других объектов.
4. Каждый объект является представителем класса, который выражает общие свойства объектов (таких, как целые числа или списки).
5. В классе задаётся поведение (функциональность) объекта. Тем самым все объекты, которые являются экземплярами одного класса, могут выполнять одни и те же действия.
6. Классы организованы в единую древовидную структуру с общим корнем, называемую иерархией наследования. Память и поведение, связанное с экземплярами определённого класса, автоматически доступны любому классу, расположенному ниже в иерархическом дереве.

Особенностями данных объектно-ориентированных языков использование механизмов, таких как – наследование, инкапсуляция, полиморфизм.

Наследование - Создание нового класса объектов путём добавления новых элементов (методов). Некоторые объектно-ориентированные языки позволяют выполнять множественное наследование, то есть объединять в одном классе возможности нескольких других классов.

Инкапсуляция – сокрытие деталей реализации, которое позволяет вносить изменения в части программы безболезненно для других её частей, что существенно упрощает сопровождение и модификацию ПО.

Полиморфизм – при полиморфизме некоторые части (методы) родительского класса заменяются новыми, реализующими специфические для данного потомка действия. Таким образом, интерфейс классов остаётся прежним, а реализация методов с одинаковым названием и набором параметров различается. В ООП обычно применяется полиморфизм подтипов (называемый при этом просто «полиморфизмом»), нередко в форме позднего связывания.

Абстракция – для выделения в моделируемом предмете важного для решения конкретной задачи по предмету, в конечном счёте — контекстное понимание предмета, формализуемое в виде класса;

Многие современные языки созданы специально для облегчения объектно-ориентированного программирования. Техники объектно-ориентированного программирования можно использовать и для не-объектно-ориентированного программирования. Такие языки содержат следующий набор элементов:

* объявление классов с полями (данными — членами класса) и методами (функциями — членами класса).
* механизм расширения класса (наследования) — порождение нового класса от существующего с автоматическим включением всех особенностей реализации класса-предка в состав класса-потомка. Большинство ООЯ поддерживают только единичное наследование.
* полиморфные переменные и параметры функций (методов), позволяющие присваивать одной и той же переменной экземпляры различных классов.
* полиморфное поведение экземпляров классов за счёт использования виртуальных методов. В некоторых ООЯ все методы классов являются виртуальными.

Некоторые языки также добавляют дополнительные средства, такие как:

* конструкторы, деструкторы, финализаторы;
* свойства (аксессоры);
* индексаторы;
* средства управления видимостью компонентов классов (интерфейсы или модификаторы доступа, такие как public, private, protected, feature и др.).

Не все языки отвечают принципам ООП в полной мере, существуют гибридные языки объединяющие объективную парадигму с другой парадигмой или даже двумя и более.

Всё это позволяет проводить разработку крупных программных комплексов командой программистов, что позволяет выполнять задачи объединения и создания компонентов выполняются разными людьми в случае необходимости. Одним из примеров подобный языков является Python.

Python как актуальный язык для написания приложений

Python является высокоуровневым языком программирования общего назначения, основной характеристикой языка являются повышенная производительность разработки и читаемость написанного кода. Python обладает минималистичным синтаксисом.

Язык поддерживает различные методологии, такие как – структурная, обобщенная, объектно-ориентированная, аспектно-ориентированная и. функциональная.

Основными архитектурными чертами являются – автоматическое управление памятью, динамическая типизация, полная интроспекция, поддержка многопоточный вычислений, высокоуровневые структуры данных, механизм обработки исключений, разбиение программ на модули, объединение модулей в пакеты.

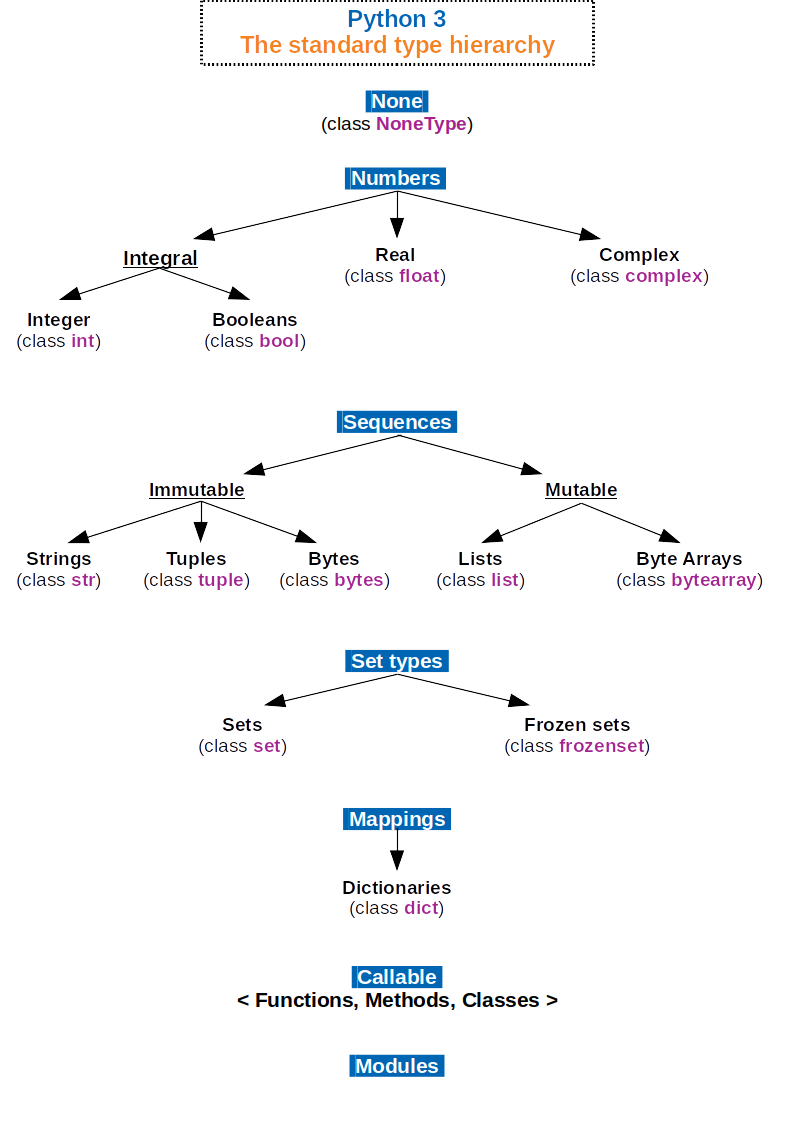
Python придерживается определенной философии сформулированной Тимом Петерсом, звучит она так:

* красивое лучше, чем уродливое;
* явное лучше, чем неявное;
* простое лучше, чем сложное;
* сложное лучше, чем запутанное;
* плоское лучше, чем вложенное;
* разреженное лучше, чем плотное;
* читаемость имеет значение;
* особые случаи не настолько особые, чтобы нарушать правила;
* при этом практичность важнее безупречности;
* ошибки никогда не должны замалчиваться;
* если не замалчиваются явно;
* встретив двусмысленность, отбрось искушение угадать;
* должен существовать один — и, желательно, только один — очевидный способ сделать это;
* хотя он поначалу может быть и не очевиден, если вы не голландец;
* сейчас лучше, чем никогда;
* хотя никогда зачастую лучше, чем прямо сейчас;
* если реализацию сложно объяснить — идея плоха;
* если реализацию легко объяснить — идея, возможно, хороша;
* пространства имён — отличная вещь! Давайте будем делать их больше!

Данный текст можно вывести при помощи команды import this.

Python работает почти на всех платформах, начиная от КПК и заканчивая мейнфреймами. Возможности портируемости обеспечиваются поддержкой характерных для каждой платформы технологий, например, для виртуальных машин Java существует версия Python под название Jython, что позволяет выполнять работу в интерпретаторе на любой системе с поддержкой Java.

Python поддерживает динамическую типизацию, вследствие чего тип переменной определяется лишь во время исполнения. В Python имеются определенные встроенные типы (рисунок 1). Каждое значение является объектом, в том числе функции, модули, классы и методы. Добавление новых типов возможно посредству написания классов с поддержкой наследования и метапрограммирования.



Стандартная иерархия типов в Python.

Синтаксис языка является четким и последовательным, что упрощает чтение кода человеком. В терминологии с 2018 года в коде при передаче аргументов в функции используются parent (родитель) и child (ребенок).

Набор операторов в Python состоит из:

* if (если) – условный оператор. Альтернативным блоком является else (иначе). При нескольких альтернативах используется elif;
* while (пока) и for (для) – операторы цикла. Для прерывания цикла и перехода к следующей итерации возможно использовать break (разрыв) и continue (продолжить);
* class (класс) – оператор определения класса;
* def (определить) – оператор определения функции, метода или генератора. Для возврата из функции или метода можно использовать оператор return (возврат), или yield (давать) в случае с генератором;
* try – except – else (пробовать – исключить – иначе) или try – finally (пробовать – заключение) – оператор обработки исключений;
* pass (пропустить) – оператор ничего не делает, применяется для пустых блоков кода.

Блоки в Python выделяются при помощи табуляции, что приучает программистов к более чистому стилю написания кода и сокращает количество строк кода.

Выражения являются полноправными операторами в Python. Синтаксис, состав, приоритет и ассоциативность операций являются стандартными для языков программирования.

Имена могут начинаться с любой латинской буквы, или буквы любого алфавита Unicode начиная с Python 3, любого регистра или подчеркивания, после чего можно указывать и цифры, но нельзя использовать ключевые слова. Интерпретатор имеет доступ к трём пространствам имен: локальному, глобальному и встроенному.

## Разработка приложения ‘помощник для начинающий прекупов’.

## Анализ существующих решений:

Многие начинающие предприниматели хотят успешно начать зарабатывать в сфере перепродажи авто .В ходе работы они могут столкнуться с многочисленными проблемами и вопросами Поэтому я решил создать приложение ,которое сможет им с различными неудачами в ходе перепродажи .Данный вид экономический деятельность очень популярен в нынешнее время и имеет большую доходность при малых вложениях.

**Разберем суть этого приложения:**

Пользователь запускает приложение .Открывается окно с главным окном с 10 кнопками разделов , при нажатии которых выводится информация данного раздела и кнопка выхода из приложения.

**Меню состоит из:**

1 кнопка “ Личные качества перекупа”

2 кнопка “ С чего начать бизнес по перепродаже авто”

3 кнопка “ Юридическая грамотность”

4 кнопка “ Приложения и сервисы для перекупов ”

5 кнопка “ Цены на авто”

6 кнопка “ Переговоры с клиентами”

7 кнопка “ Как стать перекупщиком авто”

8 кнопка “ Предпродажная подготовка”

9 кнопка “ Продажа авто перекупом”

10 кнопка “ Выход”

**На основе выполненного анализа существующих аналогов приложений можно приступить к составлению требований к создаваемому приложению.**

**1.Составление требований к приложению**

**(ТЕХНИЧЕСКОЕ ЗАДАНИЕ)**

**Каждое приложение должно соответствовать определенным требованиям, которые обоснованы задачами, выполняемыми приложением:**

**Наименование программы:** ”Помощник для начинающим перекупом”**;**

**Основание для разработки:** Углубление знаний в языке программирования Python. Создание приложения “Помощник для начинающих перекпов”.

**Назначение разработки:** Создание программного средства для начала экономической деятельности в сфере авто перепродаж.

**Назначение и область применения программы:** Предоставление

пользователю полного функционала данной приложения. Программа должна использоваться на любых объектах заказчика.

**Требования к программе:**

1)Приложение “Помощник для начинающих перекупов”

2)Горячие клавиши:

1.esc-выход

3)Изменение состояние кнопок после нажатия.

4)Графический интер фейс должен быть доступным для понимания пользователем и корректно отображать кнопки.

5)Приложение должно иметь свое имя и уникальную иконку.

**Спецификация качества программного обеспечения:**

Производительность: Программа должна предъявлять требования к

аппаратным ресурсам, требования к аппаратному обеспечению намного

ниже, чем для самой операционной системы.

Практичность: Программа должна иметь понятный пользовательский графический интерфейс.

Надёжность: Программа должна являться автономной.

Функциональность: Программа должна являться полностью завершенным программным обеспечением.

**Требования к программной документации:** Документация прилагается.

**Стадии и этапы разработки:**

-Формирование технического задания

-Проектирование

-Разработка

-Тестирование

-Написание руководства по эксплуатации

**2.Проектирование программы**

На данном этапе исследуется структура и взаимосвязи элементов разрабатываемой системы. Результатом этого этапа является проект ,содержащий достаточное количество информации для реализации системы.

**Проектирование разделится на:**

1. Проектирование архитектуры приложения

2. Проектирование пользовательского интерфейса

**2.1. Проектирование архитектуры приложения**

Обработка нажатых кнопок и клавиш через функцию def.

Что позволит достичь полной функциональности приложения <Помощник для начинающих перекупов>.

Создание анимации с помощью смены изображений на созданном окне.

Смена состояния нажатых кнопок и изменение значений самой кнопки и суммы нажатых кнопок.

Постоянное обновление элементов окна(скрытие и появление)

**2.2Проектирование пользовательского интерфейса**

Так будет выглядеть главная форма моего программного средства.

**Она будет содержать несколько элементов**:

1 кнопка “ Личные качества перекупа”

2 кнопка “ С чего начать бизнес по перепродаже авто”

3 кнопка “ Юридическая грамотность”

4 кнопка “ Приложения и сервисы для перекупов ”

5 кнопка “ Цены на авто”

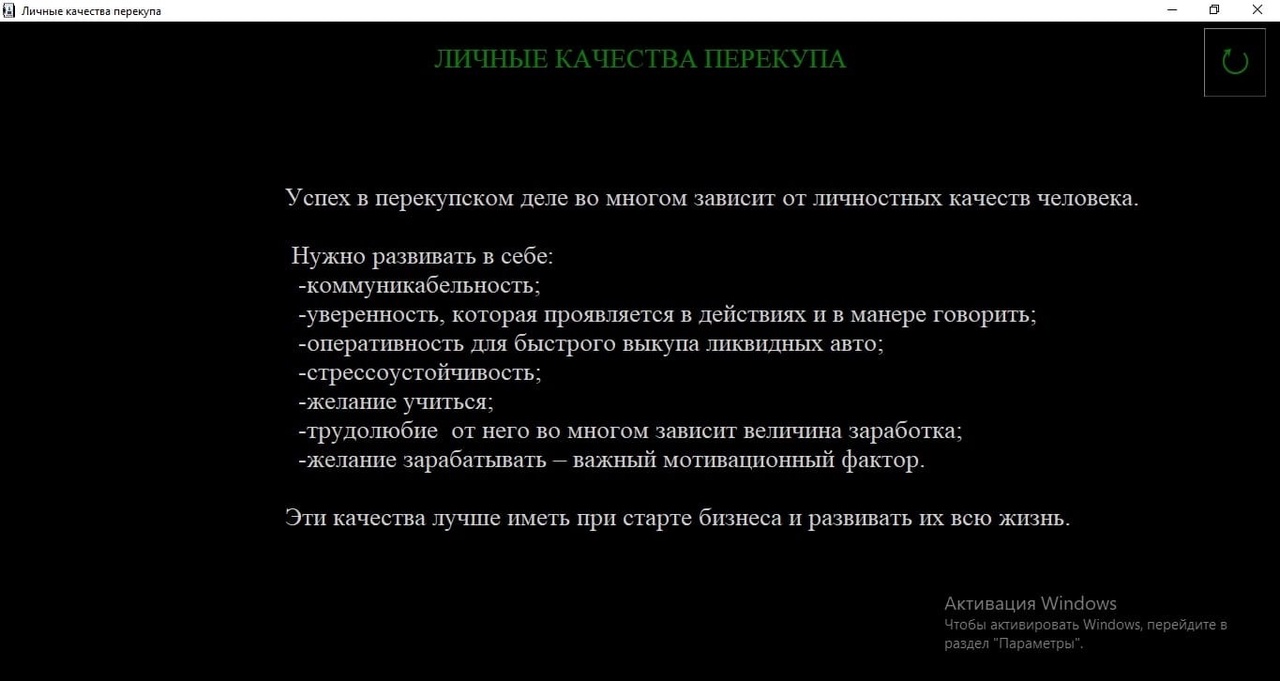
6 кнопка “ Переговоры с клиентами”

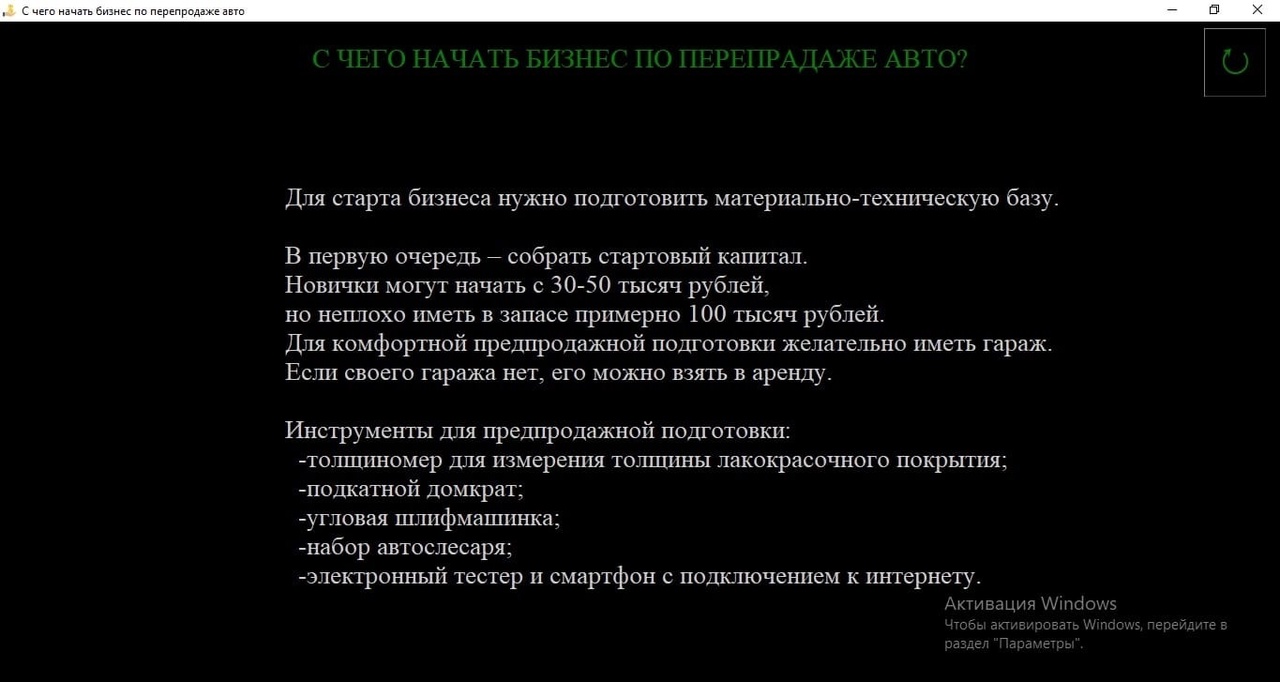
7 кнопка “ Как стать перекупщиком авто”

8 кнопка “ Предпродажная подготовка”

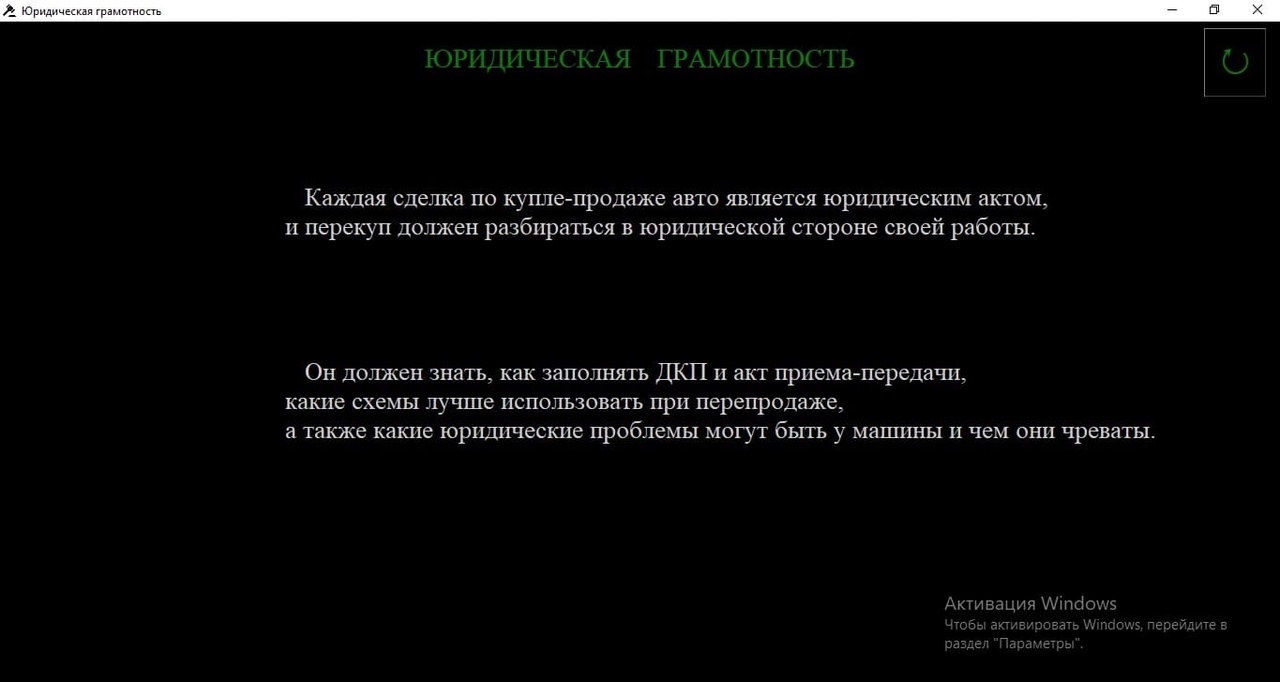
9 кнопка “ Продажа авто перекупом”

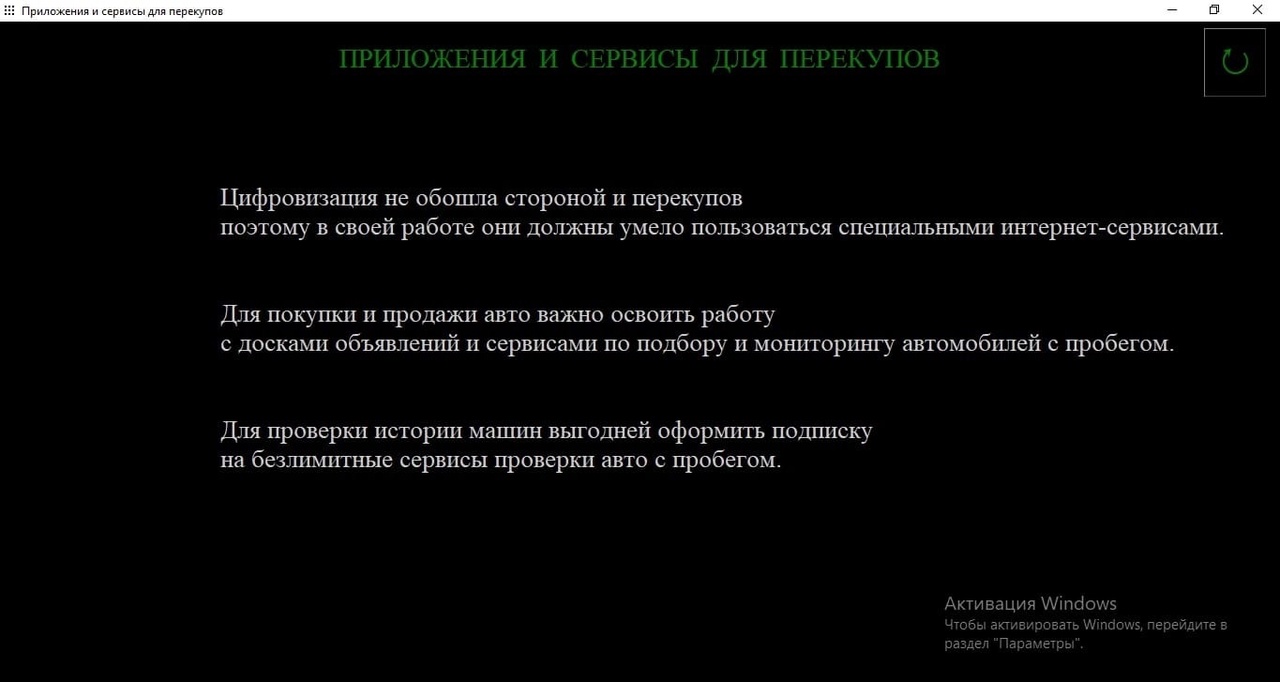
10 кнопка “ Выход”

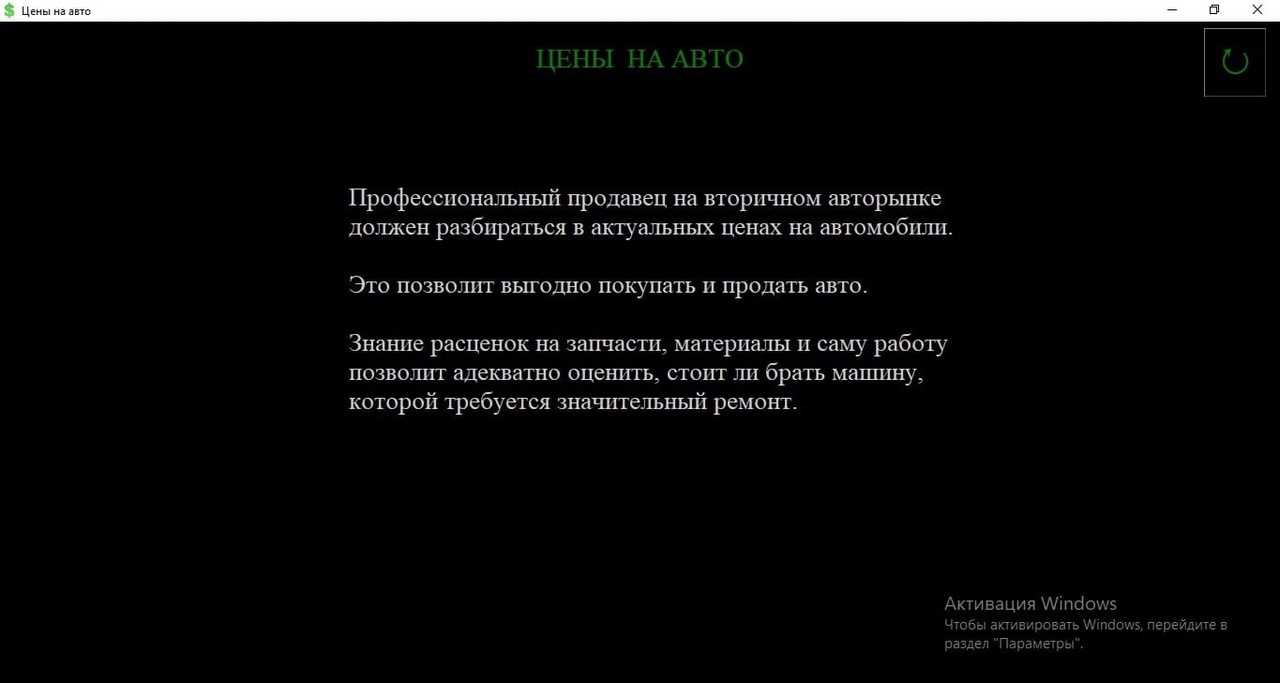
При нажатии кнопки “ Личные качества перекупа” на экран выводится :

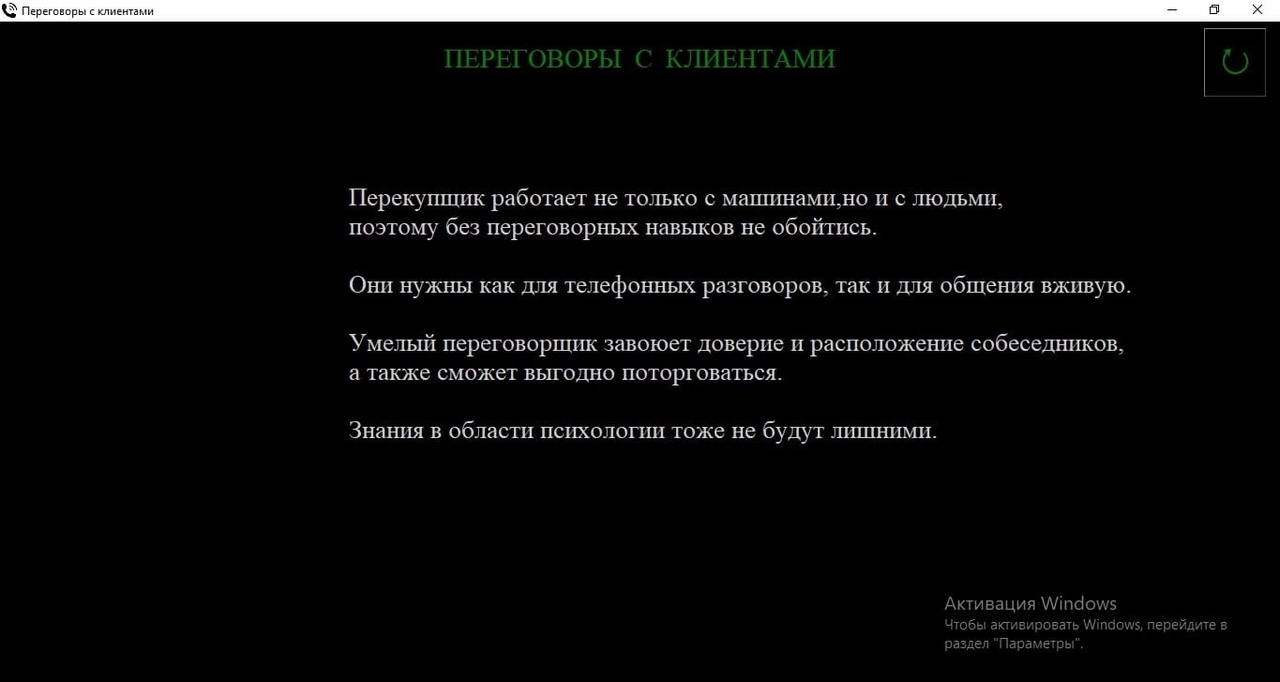
При нажатии кнопки “ С чего начать бизнес по перепродаже авто”:

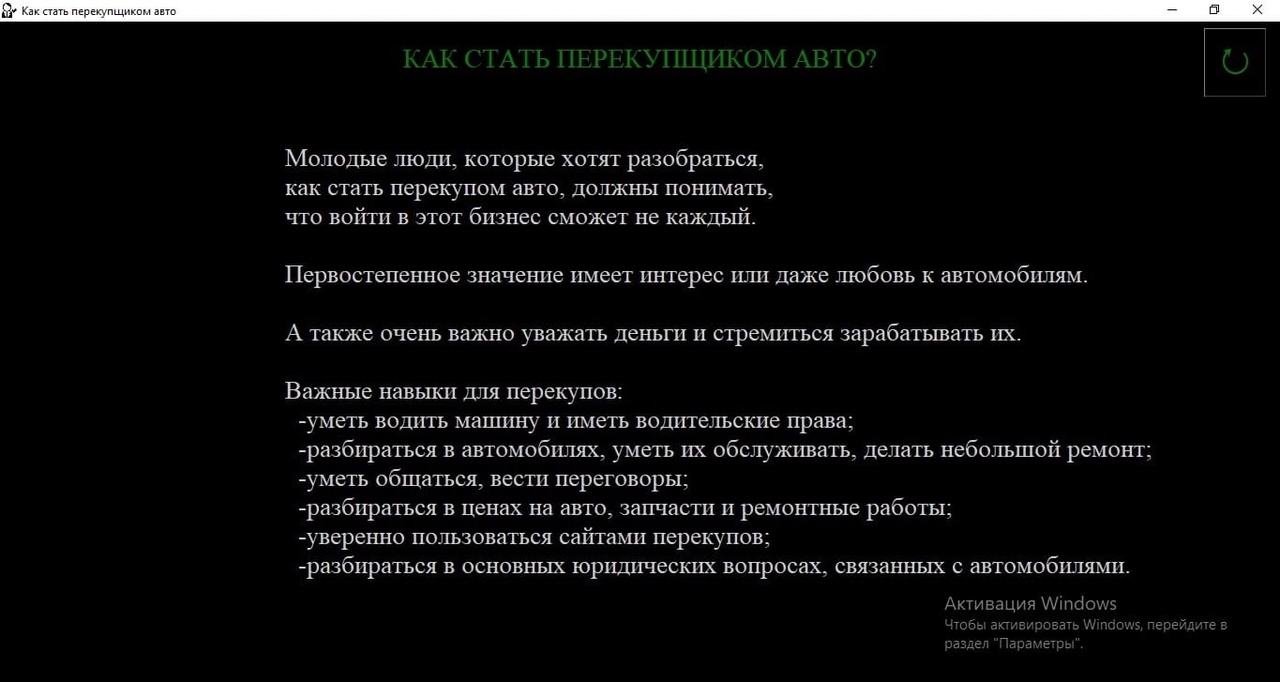
При нажатии кнопки “ Юридическая грамотность”:

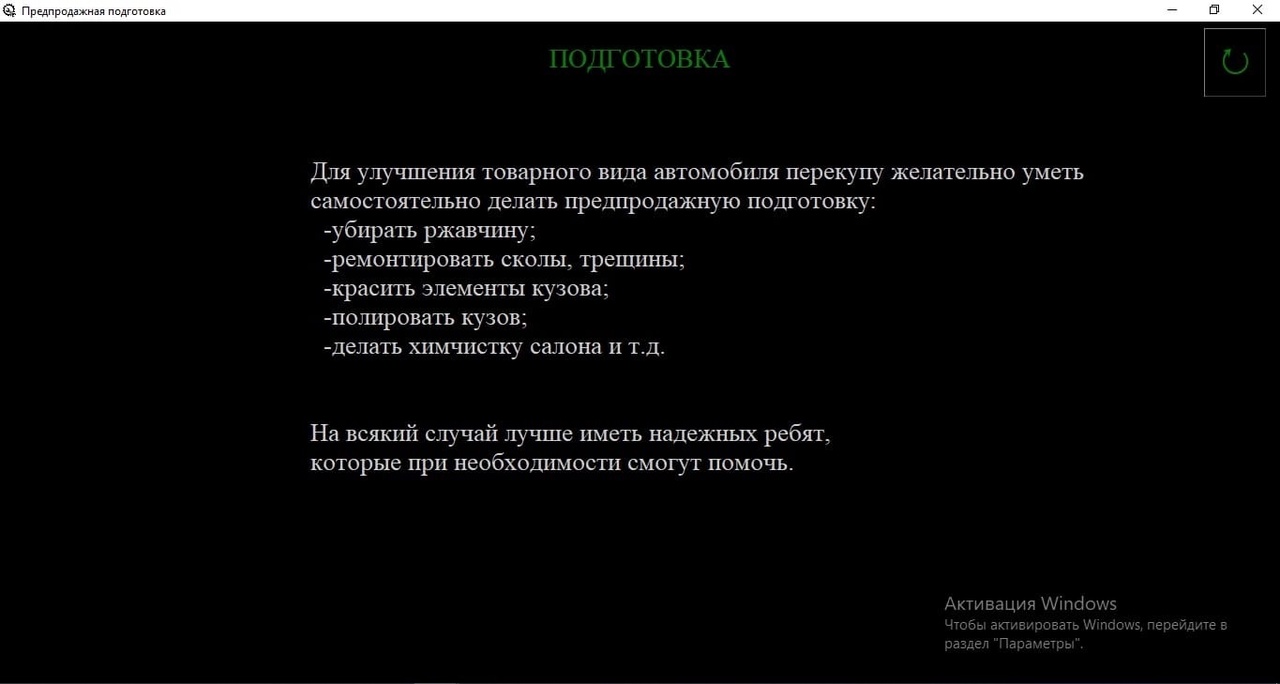


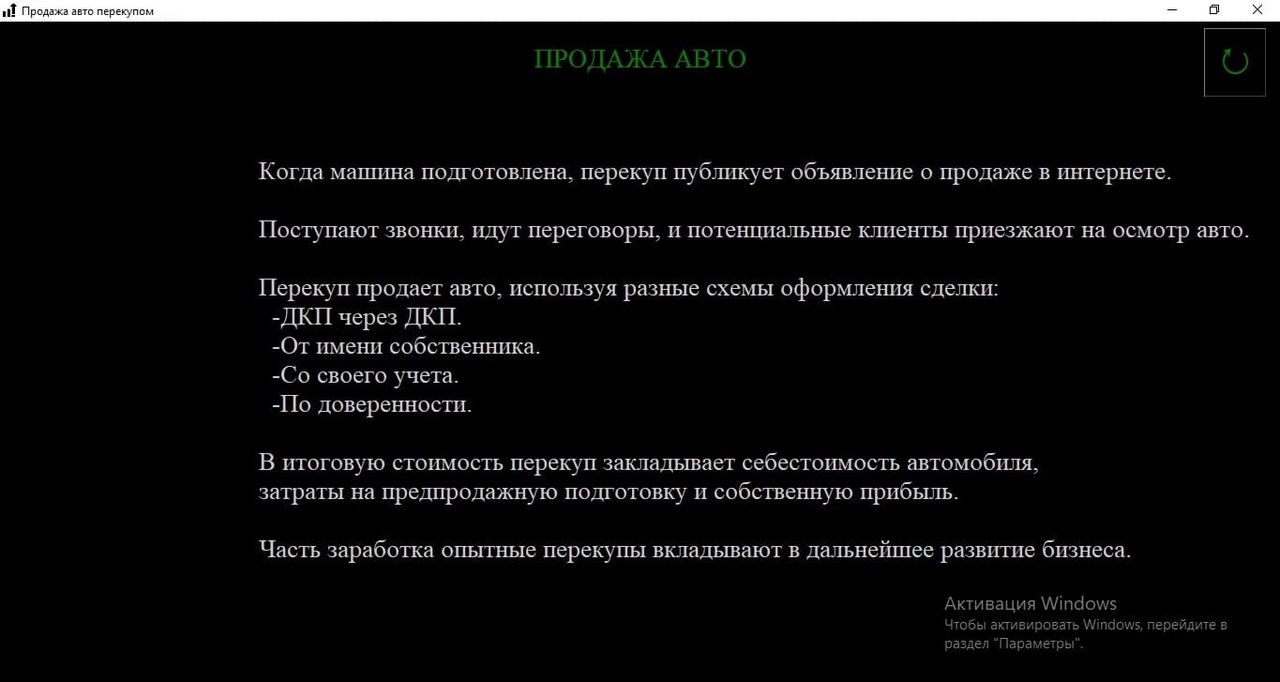
При нажатии кнопки “ Приложения и сервисы для перекупов”:

При нажатии кнопки “ Цены на авто”:

При нажатии кнопки “ Переговоры с клиентами”:

При нажатии кнопки “ Как стать перекупщиком авто”:

При нажатии кнопки “ Предпродажная подготовка”:

При нажатии кнопки “ Продажа авто перекупом”:

При нажатии кнопки “ Выход” приложение закрывается:

**3. Разработка программы**

**На данном этапе происходит разработка загрузчика и соответственно отладка программы. Разработка разделится на:**

1. Установка Python

2. Установка необходимых библиотек

3. Описание алгоритма на естественном языке

4. Реализация кода программы

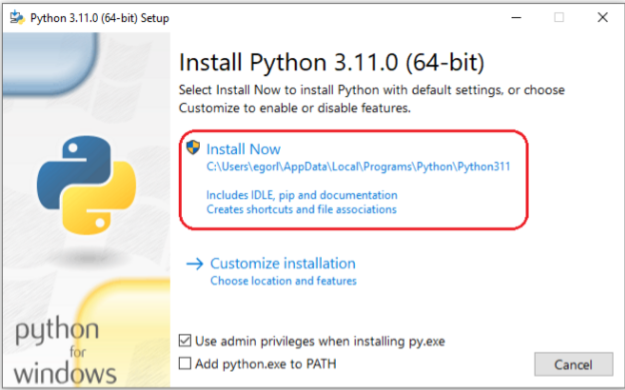
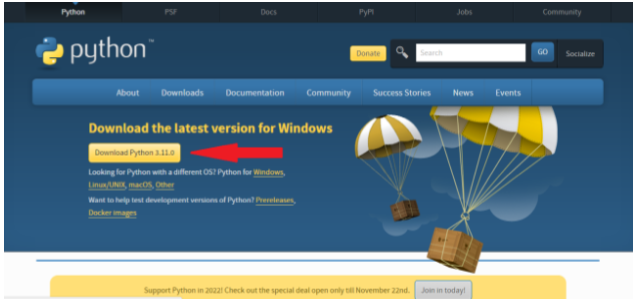
5. Листинг кода

**3.1. Установка Python**

Для создания программ на Python нам необходим интерпретатор. Для его

установки необходимо перейти на официальную страницу

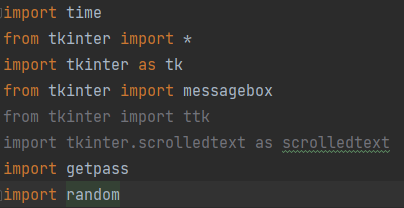
https://www.python.org/downloads/ и скачать последнюю версию языка Python.После нажатия кнопки установщик загрузится на вашу ОС. Запускаем установщик.



У нас появится окно мастера установки. После установки мы можем проверить установку Python и его версию через терминал/командную строку прописав команду python –version.

**3.2. Установка необходимых библиотек**

Для установки необходимых библиотек необходимо прописать несколько

простых команд. Ниже представлены сами команды**.**

**pip install tk**

**3.3. Описание алгоритма на естественном языке.**

После запуска приложения появится главный экран на котором представлен фон с логотипом и 10 кнопок .После нажатий которых открывается данный раздел с информацией.

**3.4 Реализация кода программы**

Написание кода программы реализовывалось в течении 15 дней. Код моего

проекта реализовывался в написании функций для кнопок.

Сначала было необходимо создать графический интерфейс приложения. После чего начал писать функции к каждой кнопке и обрабатывать их.Код программы содержит около 250 строк.

from tkinter import \*

import tkinter as tk

from tkinter import messagebox

from tkinter import ttk

#переменные цвета фона и шрифта для главного окна

fon="#3e3e3e"

shrift="white"

#выполнение действий при нажататия на 1 кнопку

def lu4Hble\_ka4ecTBa\_nepekyna\_1\_fun():

lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee= tk.Toplevel(window) # создание дочернего окна после нажатия на кнопку

lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee.geometry('1366x768') #разрешение дочернего окна

lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee.config(bg="black") #фон дочернего окна

lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee.title('Личные качества перекупа') #название дочернего окна

lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee.iconbitmap('C:\\Users\\user\\Downloads\\1.ico') # иконка дочернего окна

nazad\_button = Button(lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee, text=f"↻", fg="green", bg="black", font='Times 28',command=lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee.destroy) # кнопка на дочернем окне которая разрушает дочернее окно и возвращает в главное окно

nazad\_button.place(rely=0.01, relx=0.94) # размещение

label\_zagolivok1 = Label(lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee, text=f"ЛИЧНЫЕ КАЧЕСТВА ПЕРЕКУПА", bg="black",fg="green", font='Times 22') # заголовок в дочернем окне

label\_zagolivok1.pack(pady=20) # размещение

text = "Успех в перекупском деле во многом зависит от личностных качеств человека.\n\n Нужно развивать в себе:\n -коммуникабельность;\n -уверенность, которая проявляется в действиях и в манере говорить;\n -оперативность для быстрого выкупа ликвидных авто;\n -стрессоустойчивость;\n -желание учиться;\n -трудолюбие от него во многом зависит величина заработка;\n -желание зарабатывать – важный мотивационный фактор.\n\nЭти качества лучше иметь при старте бизнеса и развивать их всю жизнь." #основной текст дочернего окна

label\_text1 = Label(lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee, text=f"{text}", bg="black", fg="#CCCCCC", font='Times 20', width=110,anchor='w', justify=LEFT) # вывод главного текста в дочернее окно

label\_text1.place(relx=0.22, rely=0.24) #размещение

lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee.bind('<Escape>', lambda e: lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee.destroy())

lu4Hble\_ka4ecTBa\_nepekyna\_1\_docernee.state("zoomed") #растянуть на весь экран

# выполнение действий при нажататия на 2 кнопку

def na4aJlo\_6u3Heca\_2\_fun():

na4aJlo\_6u3Heca\_2\_docernee = tk.Toplevel(window)

na4aJlo\_6u3Heca\_2\_docernee.geometry('1366x768')

na4aJlo\_6u3Heca\_2\_docernee.config(bg="black")

na4aJlo\_6u3Heca\_2\_docernee.title('С чего начать бизнес по перепродаже авто')

na4aJlo\_6u3Heca\_2\_docernee.iconbitmap('C:\\Users\\user\\Downloads\\2.ico') # ИКОНКА ПРИЛОЖЕНИЯ

nazad\_button = Button(na4aJlo\_6u3Heca\_2\_docernee, text=f"↻", fg="green", bg="black", font='Times 28',command=na4aJlo\_6u3Heca\_2\_docernee.destroy)

nazad\_button.place(rely=0.01, relx=0.94)

label\_zagolivok2 = Label(na4aJlo\_6u3Heca\_2\_docernee, text=f"C ЧЕГО НАЧАТЬ БИЗНЕС ПО ПЕРЕПРАДАЖЕ АВТО?", bg="black",fg="green", font='Times 22')

label\_zagolivok2.pack(pady=20)

text = "Для старта бизнеса нужно подготовить материально-техническую базу.\n\nВ первую очередь – собрать стартовый капитал.\nНовички могут начать с 30-50 тысяч рублей,\nно неплохо иметь в запасе примерно 100 тысяч рублей.\nДля комфортной предпродажной подготовки желательно иметь гараж.\nЕсли своего гаража нет, его можно взять в аренду.\n\nИнструменты для предпродажной подготовки:\n -толщиномер для измерения толщины лакокрасочного покрытия;\n -подкатной домкрат;\n -угловая шлифмашинка;\n -набор автослесаря;\n -электронный тестер и смартфон с подключением к интернету."

label\_text2 = Label(na4aJlo\_6u3Heca\_2\_docernee, text=f"{text}", bg="black", fg="#CCCCCC", font='Times 20',width=110, anchor='w', justify=LEFT)

label\_text2.place(relx=0.22, rely=0.24)

na4aJlo\_6u3Heca\_2\_docernee.bind('<Escape>', lambda e: na4aJlo\_6u3Heca\_2\_docernee.destroy())

na4aJlo\_6u3Heca\_2\_docernee.state("zoomed")

# выполнение действий при нажататия на 3 кнопку

def ur\_rpaMoTHocTb\_3\_fun():

ur\_rpaMoTHocTb\_3\_docernee = tk.Toplevel(window)

ur\_rpaMoTHocTb\_3\_docernee.geometry('1366x768')

ur\_rpaMoTHocTb\_3\_docernee.config(bg="black")

ur\_rpaMoTHocTb\_3\_docernee.title('Юридическая грамотность')

ur\_rpaMoTHocTb\_3\_docernee.iconbitmap('C:\\Users\\user\\Downloads\\3.ico') # ИКОНКА ПРИЛОЖЕНИЯ

nazad\_button = Button(ur\_rpaMoTHocTb\_3\_docernee, text=f"↻", fg="green", bg="black", font='Times 28',command=ur\_rpaMoTHocTb\_3\_docernee.destroy)

nazad\_button.place(rely=0.01, relx=0.94)

label\_zagolivok3 = Label(ur\_rpaMoTHocTb\_3\_docernee, text=f"ЮРИДИЧЕСКАЯ ГРАМОТНОСТЬ", bg="black",fg="green", font='Times 22')

label\_zagolivok3.pack(pady=20)

text = " Каждая сделка по купле-продаже авто является юридическим актом,\nи перекуп должен разбираться в юридической стороне своей работы.\n\n\n\n\n Он должен знать, как заполнять ДКП и акт приема-передачи,\nкакие схемы лучше использовать при перепродаже,\nа также какие юридические проблемы могут быть у машины и чем они чреваты."

label\_text3 = Label(ur\_rpaMoTHocTb\_3\_docernee, text=f"{text}", bg="black", fg="#CCCCCC", font='Times 20',width=110, anchor='w', justify=LEFT)

label\_text3.place(relx=0.22, rely=0.24)

ur\_rpaMoTHocTb\_3\_docernee.bind('<Escape>', lambda e: ur\_rpaMoTHocTb\_3\_docernee.destroy())

ur\_rpaMoTHocTb\_3\_docernee.state("zoomed")

# выполнение действий при нажататия на 4 кнопку

def cepBucbl\_nepekyna\_4\_fun():

cepBucbl\_nepekyna\_4\_docernee = tk.Toplevel(window)

cepBucbl\_nepekyna\_4\_docernee.geometry('1366x768')

cepBucbl\_nepekyna\_4\_docernee.config(bg="black")

cepBucbl\_nepekyna\_4\_docernee.title('Приложения и сервисы для перекупов')

cepBucbl\_nepekyna\_4\_docernee.iconbitmap('C:\\Users\\user\\Downloads\\4.ico') # ИКОНКА ПРИЛОЖЕНИЯ

nazad\_button = Button(cepBucbl\_nepekyna\_4\_docernee, text=f"↻", fg="green", bg="black", font='Times 28',command=cepBucbl\_nepekyna\_4\_docernee.destroy)

nazad\_button.place(rely=0.01, relx=0.94)

label\_zagolivok4 = Label(cepBucbl\_nepekyna\_4\_docernee, text=f"ПРИЛОЖЕНИЯ И СЕРВИСЫ ДЛЯ ПЕРЕКУПОВ", bg="black", fg="green",font='Times 22')

label\_zagolivok4.pack(pady=20)

text = "Цифровизация не обошла стороной и перекупов\nпоэтому в своей работе они должны умело пользоваться специальными интернет-сервисами.\n\n\nДля покупки и продажи авто важно освоить работу\nс досками объявлений и сервисами по подбору и мониторингу автомобилей с пробегом.\n\n\nДля проверки истории машин выгодней оформить подписку\nна безлимитные сервисы проверки авто с пробегом."

label\_text4 = Label(cepBucbl\_nepekyna\_4\_docernee, text=f"{text}", bg="black", fg="#CCCCCC", font='Times 20', width=110,anchor='w', justify=LEFT)

label\_text4.place(relx=0.17, rely=0.24)

cepBucbl\_nepekyna\_4\_docernee.bind('<Escape>', lambda e: cepBucbl\_nepekyna\_4\_docernee.destroy())

cepBucbl\_nepekyna\_4\_docernee.state("zoomed")

# выполнение действий при нажататия на 5 кнопку

def ueHbl\_Ha\_aBTo\_5\_fun():

ueHbl\_Ha\_aBTo\_5\_docernee = tk.Toplevel(window)

ueHbl\_Ha\_aBTo\_5\_docernee.geometry('1366x768')

ueHbl\_Ha\_aBTo\_5\_docernee.config(bg="black")

ueHbl\_Ha\_aBTo\_5\_docernee.title('Цены на авто')

ueHbl\_Ha\_aBTo\_5\_docernee.iconbitmap('C:\\Users\\user\\Downloads\\5.ico') # ИКОНКА ПРИЛОЖЕНИЯ

nazad\_button = Button(ueHbl\_Ha\_aBTo\_5\_docernee, text=f"↻", fg="green", bg="black", font='Times 28',command=ueHbl\_Ha\_aBTo\_5\_docernee.destroy)

nazad\_button.place(rely=0.01, relx=0.94)

label\_zagolivok5 = Label(ueHbl\_Ha\_aBTo\_5\_docernee, text=f"ЦЕНЫ НА АВТО", bg="black",fg="green", font='Times 22')

label\_zagolivok5.pack(pady=20)

text = "Профессиональный продавец на вторичном авторынке\nдолжен разбираться в актуальных ценах на автомобили.\n\nЭто позволит выгодно покупать и продать авто.\n\nЗнание расценок на запчасти, материалы и саму работу\nпозволит адекватно оценить, стоит ли брать машину,\nкоторой требуется значительный ремонт."

label\_text5 = Label(ueHbl\_Ha\_aBTo\_5\_docernee, text=f"{text}", bg="black", fg="#CCCCCC", font='Times 20',width=110, anchor='w', justify=LEFT)

label\_text5.place(relx=0.27, rely=0.24)

ueHbl\_Ha\_aBTo\_5\_docernee.bind('<Escape>', lambda e: ueHbl\_Ha\_aBTo\_5\_docernee.destroy())

ueHbl\_Ha\_aBTo\_5\_docernee.state("zoomed")

# выполнение действий при нажататия на 6 кнопку

def neperoBopbl\_c\_kJlueHTaMu\_6\_fun():

neperoBopbl\_c\_kJlueHTaMu\_6\_docernee = tk.Toplevel(window)

neperoBopbl\_c\_kJlueHTaMu\_6\_docernee.geometry('1366x768')

neperoBopbl\_c\_kJlueHTaMu\_6\_docernee.config(bg="black")

neperoBopbl\_c\_kJlueHTaMu\_6\_docernee.title('Переговоры с клиентами')

neperoBopbl\_c\_kJlueHTaMu\_6\_docernee.iconbitmap('C:\\Users\\user\\Downloads\\6.ico') # ИКОНКА ПРИЛОЖЕНИЯ

nazad\_button = Button(neperoBopbl\_c\_kJlueHTaMu\_6\_docernee, text=f"↻", fg="green", bg="black", font='Times 28',command=neperoBopbl\_c\_kJlueHTaMu\_6\_docernee.destroy)

nazad\_button.place(rely=0.01, relx=0.94)

label\_zagolivok6 = Label(neperoBopbl\_c\_kJlueHTaMu\_6\_docernee, text=f"ПЕРЕГОВОРЫ С КЛИЕНТАМИ", bg="black", fg="green", font='Times 22')

label\_zagolivok6.pack(pady=20)

text = "Перекупщик работает не только с машинами,но и с людьми,\nпоэтому без переговорных навыков не обойтись.\n\nОни нужны как для телефонных разговоров, так и для общения вживую.\n\nУмелый переговорщик завоюет доверие и расположение собеседников,\nа также сможет выгодно поторговаться.\n\nЗнания в области психологии тоже не будут лишними."

label\_text6 = Label(neperoBopbl\_c\_kJlueHTaMu\_6\_docernee, text=f"{text}", bg="black", fg="#CCCCCC", font='Times 20', width=110,anchor='w', justify=LEFT)

label\_text6.place(relx=0.27, rely=0.24)

neperoBopbl\_c\_kJlueHTaMu\_6\_docernee.bind('<Escape>', lambda e: neperoBopbl\_c\_kJlueHTaMu\_6\_docernee.destroy())

neperoBopbl\_c\_kJlueHTaMu\_6\_docernee.state("zoomed")

# выполнение действий при нажататия на 7 кнопку

def kak\_cTaTb\_nepekynoM\_7\_fun():

kak\_cTaTb\_nepekynoM\_7\_docernee = tk.Toplevel(window)

kak\_cTaTb\_nepekynoM\_7\_docernee.geometry('1366x768')

kak\_cTaTb\_nepekynoM\_7\_docernee.config(bg="black")

kak\_cTaTb\_nepekynoM\_7\_docernee.title('Как стать перекупщиком авто')

kak\_cTaTb\_nepekynoM\_7\_docernee.iconbitmap('C:\\Users\\user\\Downloads\\7.ico') # ИКОНКА ПРИЛОЖЕНИЯ

nazad\_button = Button(kak\_cTaTb\_nepekynoM\_7\_docernee, text=f"↻", fg="green", bg="black", font='Times 28',command=kak\_cTaTb\_nepekynoM\_7\_docernee.destroy)

nazad\_button.place(rely=0.01, relx=0.94)

label\_zagolivok7 = Label(kak\_cTaTb\_nepekynoM\_7\_docernee, text=f"КАК СТАТЬ ПЕРЕКУПЩИКОМ АВТО?", bg="black",fg="green", font='Times 22')

label\_zagolivok7.pack(pady=20)

text = "Молодые люди, которые хотят разобраться,\nкак стать перекупом авто, должны понимать,\nчто войти в этот бизнес сможет не каждый.\n\nПервостепенное значение имеет интерес или даже любовь к автомобилям.\n\nА также очень важно уважать деньги и стремиться зарабатывать их.\n\nВажные навыки для перекупов:\n -уметь водить машину и иметь водительские права;\n -разбираться в автомобилях, уметь их обслуживать, делать небольшой ремонт;\n -уметь общаться, вести переговоры;\n -разбираться в ценах на авто, запчасти и ремонтные работы;\n -уверенно пользоваться сайтами перекупов;\n -разбираться в основных юридических вопросах, связанных с автомобилями."

label\_text7 = Label(kak\_cTaTb\_nepekynoM\_7\_docernee, text=f"{text}", bg="black", fg="#CCCCCC", font='Times 20',width=110, anchor='w', justify=LEFT)

label\_text7.place(relx=0.22, rely=0.18)

kak\_cTaTb\_nepekynoM\_7\_docernee.bind('<Escape>', lambda e: kak\_cTaTb\_nepekynoM\_7\_docernee.destroy())

kak\_cTaTb\_nepekynoM\_7\_docernee.state("zoomed")

# выполнение действий при нажататия на 8 кнопку

def nogroToBka\_8\_fun():

nogroToBka\_8\_docernee = tk.Toplevel(window)

nogroToBka\_8\_docernee.geometry('1366x768')

nogroToBka\_8\_docernee.config(bg="black")

nogroToBka\_8\_docernee.title('Предпродажная подготовка')

nogroToBka\_8\_docernee.iconbitmap('C:\\Users\\user\\Downloads\\8.ico') # ИКОНКА ПРИЛОЖЕНИЯ

nazad\_button = Button(nogroToBka\_8\_docernee, text=f"↻", fg="green", bg="black", font='Times 28',command=nogroToBka\_8\_docernee.destroy)

nazad\_button.place(rely=0.01, relx=0.94)

label\_zagolivok8 = Label(nogroToBka\_8\_docernee, text=f"ПОДГОТОВКА", bg="black",fg="green", font='Times 22')

label\_zagolivok8.pack(pady=20)

text = "Для улучшения товарного вида автомобиля перекупу желательно уметь\nсамостоятельно делать предпродажную подготовку:\n -убирать ржавчину;\n -ремонтировать сколы, трещины;\n -красить элементы кузова;\n -полировать кузов;\n -делать химчистку салона и т.д.\n\n\nНа всякий случай лучше иметь надежных ребят,\nкоторые при необходимости смогут помочь."

label\_text8 = Label(nogroToBka\_8\_docernee, text=f"{text}", bg="black", fg="#CCCCCC", font='Times 20',width=110, anchor='w', justify=LEFT)

label\_text8.place(relx=0.24, rely=0.2)

nogroToBka\_8\_docernee.bind('<Escape>', lambda e: nogroToBka\_8\_docernee.destroy())

nogroToBka\_8\_docernee.state("zoomed")

# выполнение действий при нажататия на 9 кнопку

def npodaga\_aBTo\_nepekynoM\_9\_fun():

npodaga\_aBTo\_nepekynoM\_9\_docernee = tk.Toplevel(window)

npodaga\_aBTo\_nepekynoM\_9\_docernee.geometry('1366x768')

npodaga\_aBTo\_nepekynoM\_9\_docernee.config(bg="black")

npodaga\_aBTo\_nepekynoM\_9\_docernee.title('Продажа авто перекупом')

npodaga\_aBTo\_nepekynoM\_9\_docernee.iconbitmap('C:\\Users\\user\\Downloads\\9.ico') # ИКОНКА ПРИЛОЖЕНИЯ

nazad\_button = Button(npodaga\_aBTo\_nepekynoM\_9\_docernee, text=f"↻", fg="green", bg="black", font='Times 28',command=npodaga\_aBTo\_nepekynoM\_9\_docernee.destroy)

nazad\_button.place(rely=0.01, relx=0.94)

label\_zagolivok9 = Label(npodaga\_aBTo\_nepekynoM\_9\_docernee, text=f"ПРОДАЖА АВТО", bg="black",fg="green", font='Times 22')

label\_zagolivok9.pack(pady=20)

text = "Когда машина подготовлена, перекуп публикует объявление о продаже в интернете.\n\nПоступают звонки, идут переговоры, и потенциальные клиенты приезжают на осмотр авто.\n\nПерекуп продает авто, используя разные схемы оформления сделки:\n -ДКП через ДКП.\n -От имени собственника.\n -Со своего учета.\n -По доверенности.\n\nВ итоговую стоимость перекуп закладывает себестоимость автомобиля,\nзатраты на предпродажную подготовку и собственную прибыль.\n\nЧасть заработка опытные перекупы вкладывают в дальнейшее развитие бизнеса."

label\_text9 = Label(npodaga\_aBTo\_nepekynoM\_9\_docernee, text=f"{text}", bg="black", fg="#CCCCCC", font='Times 20',width=110, anchor='w', justify=LEFT)

label\_text9.place(relx=0.2, rely=0.2)

npodaga\_aBTo\_nepekynoM\_9\_docernee.bind('<Escape>', lambda e: npodaga\_aBTo\_nepekynoM\_9\_docernee.destroy())

npodaga\_aBTo\_nepekynoM\_9\_docernee.state("zoomed")

# создание окна

window=Tk()

window.title('Перекуп авто') #название окна

window.geometry('1366x768') #разрешение

window.config(bg="black") #фон главного окна

window.protocol("WM\_DELETE\_WINDOW", window.iconify)

window.bind('<Escape>', lambda e: window.destroy()) #закрытие по esc

window.iconbitmap('C:\\Users\\ivans\\Downloads\\иконка приложения.ico') #иконка окна

# картинка

window.iconka\_po\_centry=PhotoImage(file="C:\\Users\\user\\Downloads\\фон перекупа.png") #картинка на главное окно(фон)

iconka\_po\_centry=Label(window,image=window.iconka\_po\_centry,bg="black") #окно виндовс , переменная картинки, и заполнение фона черным

iconka\_po\_centry.pack() #метод pack размещает по центру

#создание кнопки 1(текст,цвет,размер,команда) и размещение на окна с помощью метода place

lu4Hble\_ka4ecTBa\_nepekyna\_1\_button=Button(text=f"Личные качества перекупа",fg=shrift,bg=fon,font='Times 14',width=23,height=3,command=lu4Hble\_ka4ecTBa\_nepekyna\_1\_fun)

lu4Hble\_ka4ecTBa\_nepekyna\_1\_button.place(rely=0.08,relx=0.02)

#создание кнопки 2(текст,цвет,размер,команда) и размещение на окна с помощью метода place

na4aJlo\_6u3Heca\_2\_button=Button(text=f"С чего начать бизнес\nпо перепродаже авто",fg=shrift,bg=fon,font='Times 14',width=23,height=3,command=na4aJlo\_6u3Heca\_2\_fun)

na4aJlo\_6u3Heca\_2\_button.place(rely=0.27,relx=0.02)

#создание кнопки 3(текст,цвет,размер,команда) и размещение на окна с помощью метода place

ur\_rpaMoTHocTb\_3\_button=Button(text=f"Юридическая грамотность",fg=shrift,bg=fon,font='Times 14',width=23,height=3,command=ur\_rpaMoTHocTb\_3\_fun)

ur\_rpaMoTHocTb\_3\_button.place(rely=0.46,relx=0.02)

#создание кнопки 4(текст,цвет,размер,команда) и размещение на окна с помощью метода place

cepBucbl\_nepekyna\_4\_button=Button(text=f"Приложения и сервисы\n для перекупов",fg=shrift,bg=fon,font='Times 14',width=23,height=3,command=cepBucbl\_nepekyna\_4\_fun)

cepBucbl\_nepekyna\_4\_button.place(rely=0.65,relx=0.02)

#создание кнопки 5(текст,цвет,размер,команда) и размещение на окна с помощью метода place

ueHbl\_Ha\_aBTo\_5\_button=Button(text=f"Цены на авто",fg=shrift,bg=fon,font='Times 14',width=23,height=3,command=ueHbl\_Ha\_aBTo\_5\_fun)

ueHbl\_Ha\_aBTo\_5\_button.place(rely=0.84,relx=0.02)

#создание кнопки 6(текст,цвет,размер,команда) и размещение на окна с помощью метода place

neperoBopbl\_c\_kJlueHTaMu\_6\_button=Button(text=f"Переговоры с клиентами",fg=shrift,bg=fon,font='Times 14',width=23,height=3,command=neperoBopbl\_c\_kJlueHTaMu\_6\_fun)

neperoBopbl\_c\_kJlueHTaMu\_6\_button.place(rely=0.08,relx=0.79)

#создание кнопки 7(текст,цвет,размер,команда) и размещение на окна с помощью метода place

kak\_cTaTb\_nepekynoM\_7\_button=Button(text=f"Как стать\nперекупщиком авто",fg=shrift,bg=fon,font='Times 14',width=23,height=3,command=kak\_cTaTb\_nepekynoM\_7\_fun)

kak\_cTaTb\_nepekynoM\_7\_button.place(rely=0.27,relx=0.79)

#создание кнопки 8(текст,цвет,размер,команда) и размещение на окна с помощью метода place

nogroToBka\_8\_button=Button(text=f"Предпродажная подготовка",fg=shrift,bg=fon,font='Times 14',width=23,height=3,command=nogroToBka\_8\_fun)

nogroToBka\_8\_button.place(rely=0.46,relx=0.79)

#создание кнопки 9(текст,цвет,размер,команда) и размещение на окна с помощью метода place

npodaga\_aBTo\_nepekynoM\_9\_button=Button(text=f"Продажа авто перекупом",fg=shrift,bg=fon,font='Times 14',width=23,height=3,command=npodaga\_aBTo\_nepekynoM\_9\_fun)

npodaga\_aBTo\_nepekynoM\_9\_button.place(rely=0.65,relx=0.79)

#глобальный выход по кнопке и размещение на окне

exit\_global\_button=Button(text=f"Выход",fg="red",bg="black",font='Times 14',width=23,command=window.destroy,height=3)

exit\_global\_button.place(rely=0.84,relx=0.79)

window.iconka\_po\_centry1=PhotoImage(file="C:\\Users\\user\\Downloads\\инста-надпись.png") #надпись инсты

iconka\_po\_centry1=Label(window,image=window.iconka\_po\_centry1,bg="black") #окно виндовс , переменная картинки, и заполнение фона черным

iconka\_po\_centry1.pack() #метод pack размещает по центру

window.state("zoomed") #открытие окна на весь экран

window.mainloop()

**3.5. Листинг кода**

1)Подключение библиотек

2)функции каждой кнопки и обработки различных действий

3)создание окна и привязка параметров к окну

4)отслеживание горячих клавиш

5)создание кнопок привязка к ним функций и параметров

**4. Тестирование программы:**

**Тестирование прошло успешно. Все кнопки работают исправно и выводят нужную информацию.**

**5. Руководство по эксплуатации**

Необходимо скопировать программу на жесткий диск компьютера и нажать на ее значок двойным нажатием левой кнопки мыши. При нажатии откроется окно с кнопками .Нажмите на интересующий вам отдел и он выведет нужную информацию.

**Заключение**

В процессе исследования темы моей курсовой работы, я углубил свои знания в языке программирования Python. В свое работе я использовал большой функционал PyCharm и различные библиотеки . С помощью этого я смог создать алгоритм полноценного приложения “Помощник для начинающих прекупов”.

**Список источников и литературы**

1 https://www.kolesa.ru/article/kak-stat-perekupom-avtomobiley-s-nulya-i-zarabatyvat-ot-100-tysyach-rubley-v-mesyats

2 https://auto.ru

3 <https://docs.python.org/3/library/tkinter.html>

4 <https://microsoft-paint-3d.softonic.ru/>

5 <https://ru.wikipedia.org/wiki/История_языков_программирования>

6 <https://www.python.org/>

7 <https://ru.wikipedia.org/wiki/Python>

8 https://autoteka.ru

9 <https://www.nomerogram.ru>

10 <https://avtocod.ru/kak-stat-perekupom>

11 https://kulikavto.ru/kuplya-prodazha/kak-zanyatsya-pereprodazhej-avtomobilej.html