**Департамент образования и науки города Москвы**

**Государственное бюджетное общеобразовательное учреждение**

**города Москвы «Образовательный центр «Протон»**

**Покерный аналитик**

Чернов Владимир Евгеньевич

ученик 10-Т класса

ГБОУ Образовательный центр «Протон»

Научный руководитель:

Руководитель ИТ-проектов

ГБОУ Образовательный центр «Протон»

Федоров Кирилл Евгеньевич

**Москва, 2022**

Оглавление

[Введение 3](#_Toc96939961)

[Актуальность работы 3](#_Toc96939962)

[Обоснование выбора темы 3](#_Toc96939963)

[Цель и задачи работы 4](#_Toc96939964)

[Методика выполнения работы 5](#_Toc96939965)

[Анализ существующих решений 5](#_Toc96939966)

[Методы работы 5](#_Toc96939967)

[Архитектура программы 6](#_Toc96939968)

[Место и сроки выполнения работы 16](#_Toc96939969)

[Результаты 16](#_Toc96939970)

[Описание завершённого продукта 17](#_Toc96939971)

[Список использованной литературы 17](#_Toc96939972)

# **Введение**

## **Актуальность работы**

Покер – игра с колодой из пятидесяти двух карт. Довольно сложная игра, требующая большого количества логических размышлений, а также хорошей памяти. Игра относится к логически-стратегическим и, возможно, сложнее шахмат. [1]

Все приложения, посвящённые вопросу о предоставлении конкретных шансов на различных стадиях игры в «Покер», существующие ранее, были запрещены, ибо взаимодействовали с покерным клиентом, от чего их спокойно блокировал тот или иной сайт, а не принимал входные данные от самого пользователя.

Разрабатываемое приложение отличается от существующих, что оно фактически никак не взаимодействует с покерным клиентом, а лишь рассчитывает конкретные проценты, в зависимости от карт, которые у пользователя и карт, которые на столе у всех игроков.

## **Обоснование выбора темы**

Данная тема была выбрана в связи с личной заинтересованностью в решении конкретной логической задачи с получением шансов на конкретных стадиях игры в «Покер». Побочным решением является, что эти шансы фактически можно предоставлять пользователю по средству взаимодействия с графическим интерфейсом.

# **Цель и задачи работы**

**Цель работы:**

Разработать программу решения заданной ранее логической задачи, условиями которой являются конкретизированные данные на различных стадиях игры в «Покер» для того или иного пользователя.

**Задачи работы:**

1. Проанализировать существующие решения использования различных приложений в игре в «Покер» с целью получить конкретные шансы (в процентах) на различных стадиях игры.
2. Разработать программу с графическим интерфейсом, взаимодействующую с табличными значениями и базами данных. Данное решение этой задачи взаимодействует с самим пользователем и предоставляет наиболее точную информацию.
3. Провести оценку результатов на работоспособность с использованием примеров игры в «Покер».

# **Методика выполнения работы**

## **Анализ существующих решений**

Конкуренты для данной программы, скорее всего, существуют в закрытых кругах общения, в которые вряд ли можно было бы попасть очень легко и просто, а те, что и существуют, не пользуются особой популярностью, так как часть из них взаимодействуют на прямую с покерным клиентом, что будет легко отследить и отключить доступ к игре в том или ином игровом клиенте.

Представленная программа работает вне зависимости от игрового клиента, используемого пользователем.

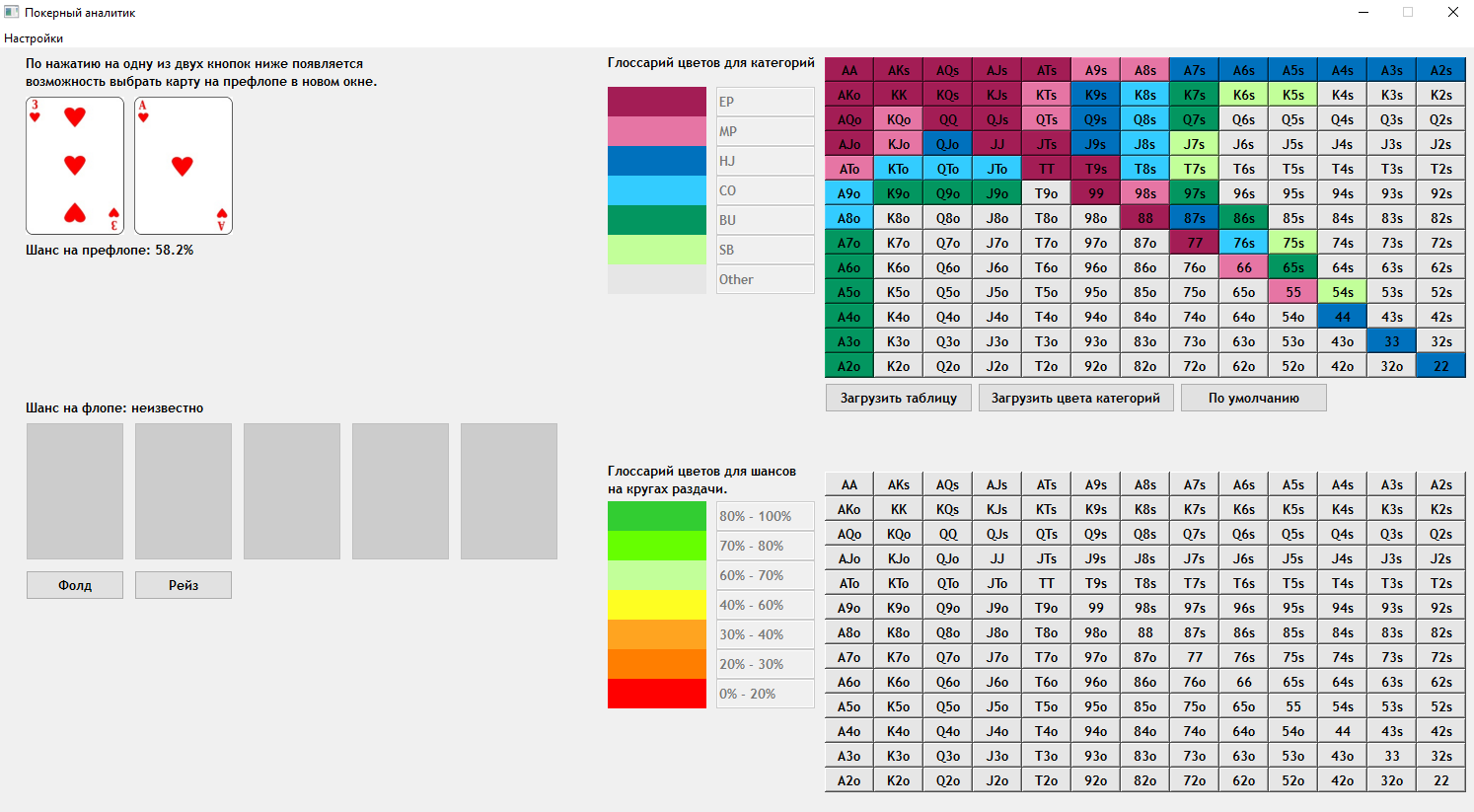
## **Методы работы**

Использование языка программирования Python, существующие библиотеки данного языка.

Использование внешних таблиц (формата «.csv») и баз данных SQLite.

## **Архитектура программы**

**Файл «ui.py» - front-end – user-interface**

Класс MainWindow, наследуемый от виджета библиотеки PyQT5.QtWidgets – QMainWindow. Данный класс является виджетом, который также является главным окном моей программы.Приложение: Рисунок 1. ****  
Рисунок 1.

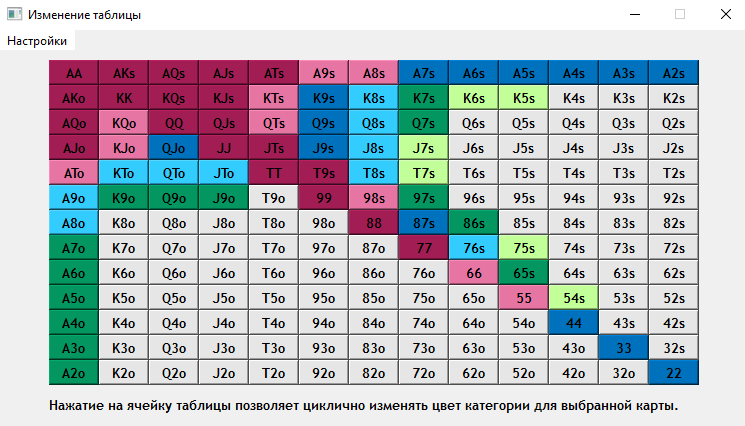
**Константы файла:**

* **SIZE\_MW –** размеры главного окна в формате списка
* **SIZE\_SW –** размеры второго окна (окна изменения категорий карт циклично) в формате писка
* **SIZE\_TW –** размеры третьего окна (окна изменения цвета категорий) в формате списка
* **SIZE\_FW –** размеры четвёртого окна (окна выбора карты) в формате списка
* **SIZE\_FFW –** размеры пятого окна(окна отображения результатов) в формате списка

**Данный класс также содержит функции (в дальнейшем методы):**

* **\_\_init\_\_ -** метод, отвечающий за инициализацию данного класса, с именованными (имена переменных говорящие) переменными
* **initUI –** основной метод класса, отвечающий за использование функций, создающих или изменяющих главное окно, а также запуск сторонних окон
* **create\_glossary\_for\_chances –** метод, создающий глоссарий соотнесения цвета к проценту на кругах раздачи после «префлопа»
* **paint\_table\_for\_chances –** метод, вызывающий покраску таблицы кнопок (QPushButton), как только появляется словарь соотнесения конкретных карт и процентов
* **create\_table\_for\_chances –** метод, создающий таблицу 13х13, содержащий кликабельные (но не привязанные к какому-либо действию) кнопки (QPushButton) для отображения шансов на кругах раздачи после «префлопа»
* **create\_choice\_on\_next\_stage –** метод, вызывающийся для создания пяти кнопок (QPushButton, привязанные к choice\_card (см. далее)), создающие выбор карт на кругах раздачи (флоп, тёрн и ривер)
* **create\_choice –** метод, вызывающийся для создания двух кнопок, создающие выбор двух основных карт пользователем
* **clear\_choice\_on\_desk –** метод для очистки матрицы 13х13 из кнопок, описанных ранее, для показывания пользователю шансов на кругах раздачи после «префлопа», а также очистки кнопок, создающихся в ходе работы метода create\_choice\_on\_next\_stage
* **create\_choice\_on\_preflop –** метод, используемый для создания кнопок (объектов QPushButton) для выбора карт пользователя
* **calculate\_the\_total\_chance –** метод, используемый для подсчёта общего шанса на кругах раздачи после «префлопа»
* **change\_dict\_of\_chances –** метод, используемый для корректировки словаря шансов под тот формат, который будет удобнее использовать в дальнейшем
* **check\_preflop –** метод проверки «префлопа», при успешном получении результата который отображает шансы на данном круге раздачи
* **define\_a\_map –** метод, возвращающий более удобный формат вида карты для дальнейшей работы (формат: {Наименование карты (пример: Ace)} {Индекс символики карты (пример: 2)})
* **choice\_card –** метод, используемый для открытия диалогового окна, а именно объекта класса FourthWindow, в котором создаётся матрица из картинок карт, нанесённых на кнопки, дабы те в свою очередь возвращались в метод, названный ранее
* **change\_icon –** метод, который позволяет менять иконку любой кнопке
* **error\_choice\_card –** метод, вызывающий окно, сообщающее об ошибке выбора карты
* **do\_table –** метод, создающий таблицу категорий пар карт по заданным ему координатам при использовании того
* **do\_glossary –** метод, создающий таблицу категорий цветов, а именно «глоссарий», по заданным ему координатам при использовании того
* **change\_glossary –** метод, изменяющий цвета в таблице категорий, то есть в «глоссарии».Данный метод вызывается при загрузке таблицы категорий цветов, что будет описан далее
* **download\_designations –** метод, используемый для загрузки таблицы с расширением «.csv» для изменения таблицы категорий пар карт. Вызывается при нажатии на соответствующую кнопку загрузки категорий
* **download\_colors\_category –** метод, используемый для загрузки таблицы с расширением «.csv» для изменения таблицы цветов категорий, то есть «глоссария». Вызывается при нажатии на соответствующую кнопку загрузки цветов категорий
* **by\_default –** изменяет таблицу категорий пар карт, а также «глоссарий» на значения по умолчанию, сохранённые в директории программы
* **change\_table –** метод, вызывающий экземпляр класса SecondWindow для создания нового окна, в котором пользователь сможет изменять циклично права категорий для каждой пары карт, а также сохранять те, загружать другие для изменения, или же работать с таблицей по умолчанию
* **change\_color\_on\_buttons –** метод, вызывающий экземпляр класса ThirdWindow для создания окна, в котором пользователь сможет лично изменять цвета для каждой из заданных программой категорий, а также сохранять те, загружать другие для изменения, или же работать с таблицей по умолчанию

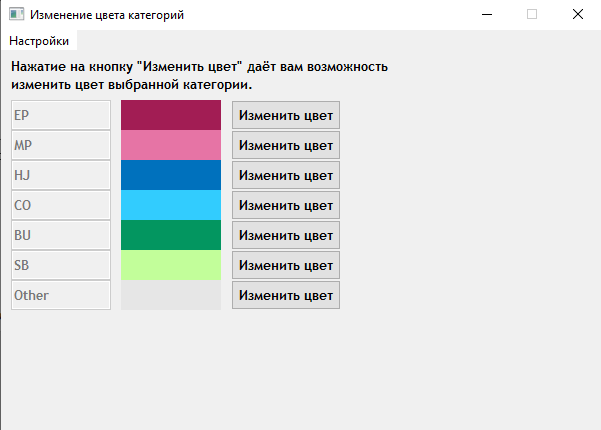
Класс SecondWindow, наследуемый от виджета библиотеки PyQT5.QtWidgets – QWidget. Данный класс является виджетом, который также является окном моей программы, для изменения пользователем цикличным образом конкретных прав категорий для выбранной пары карт.  
Приложение: Рисунок 2.

  
Рисунок 2.

**Данный класс также содержит функции (в дальнейшем методы):**

* **\_\_init\_\_ -** метод, отвечающий за инициализацию данного класса, с именованными (имена переменных говорящие) переменными
* **initUI –** основной метод класса, отвечающий за использование функций, создающих или изменяющих окно данного класса
* **open\_designations –** метод, позволяющий открывать таблицы формата «.csv», которые выберет пользователь
* **table\_by\_default –** метод, загружающий таблицу категорий «по умолчанию»
* **file\_save –** метод, позволяющий сохранять получившийся результат в таблицу формата «.csv»
* **create\_table –** метод, создающий таблицу 13х13 категорий карт из кликабельных кнопок (объекты QPushButton), в последующем привязывающиеся к функции chance\_color\_for\_button, которая позволяет циклично менять цвет
* **chance\_color\_for\_button –** метод, позволяющий привязанным объектам изменять цвет их фона циклично по нажатию

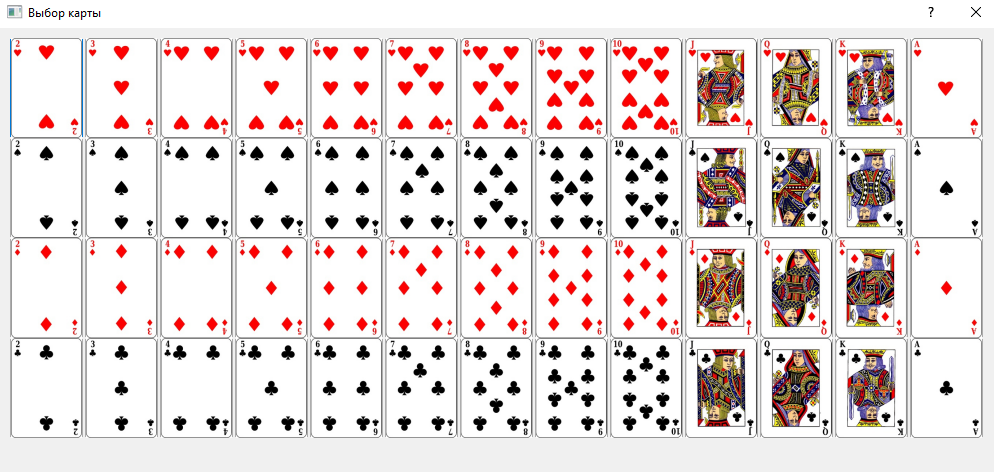
Класс ThirdWindow, наследуемый от виджета библиотеки PyQT5.QtWidgets – QWidget. Данный класс является виджетом, который также является окном моей программы, для изменения пользователем цветов под конкретные типы категорий в диалоговом окне.  
Приложение: Рисунок 3.

  
Рисунок 3.

**Данный класс также содержит функции (в дальнейшем методы):**

* **\_\_init\_\_ -** метод, отвечающий за инициализацию данного класса, с именованными (имена переменных говорящие) переменными
* **initUI –** основной метод класса, отвечающий за использование функций, создающих или изменяющих окно данного класса
* **save\_table –** метод, позволяющий сохранять получившийся результат пользователем в таблицу формата «.csv»
* **by\_default –** метод, позволяющий открыть цвета категорий «по умолчанию» из главного файла таблицы, используемого программой по умолчанию
* **open\_table\_colors –** метод, позволяющий открыть цвета категорий из какого-либо файла пользователя подходящего под формат «.csv» и содержащий цвета категорий
* **create\_table\_of\_colors –** метод, создающий таблицу цветов категорий, а также кнопки, которые вызывают диалоговые окна смены цвета
* **rgb\_to\_hex –** метод, возвращающий из значения RGB значение HEX
* **change\_color\_for\_button –** метод, исполняющий вызов диалогового окна для выбора цвета под конкретно заданную категорию, после чего конвертирует полученное значение в HEX и использует его в дальнейшем

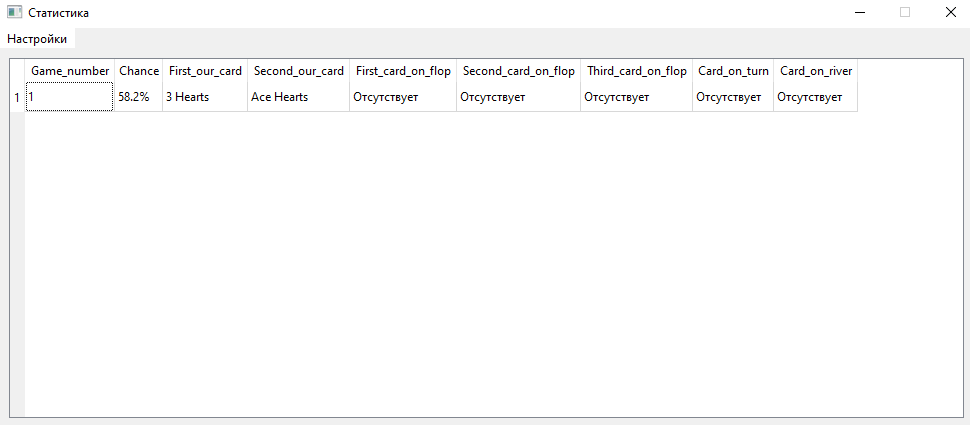
Класс FourthWindow, наследуемый от виджета библиотеки PyQT5.QtWidgets – QWidget. Данный класс является виджетом, который также является окном моей программы, для создания диалога для выбора пользователем карты под карты игрока или карты стола.  
Приложение: Рисунок 4.



**Данный класс также содержит функции:**

* **\_\_init\_\_ -** метод, отвечающий за инициализацию данного класса, с именованными (имена переменных говорящие) переменными
* **initUI –** основной метод класса, отвечающий за использование функций, создающих или изменяющих окно данного класса
* **create\_table\_of\_cards –** метод, создающий таблицу карт 4х13 кликабельных кнопок (объектов QPushButton), которые будут привязаны к функции return\_card
* **return\_card –** метод, осуществляющий закрытие данного окна и передачи значение выбранной карты в главное окно программы

Класс FifthWindow, наследуемый от виджета библиотеки PyQT5.QtWidgets – QWidget. Данный класс является виджетом, который также является окном моей программы, для отображения результатов игр в формате таблицы.  
Приложение: Рисунок 5.

  
Рисунок 5.

**Данный класс также содержит функции:**

* **\_\_init\_\_ -** метод, отвечающий за инициализацию данного класса, с именованными (имена переменных говорящие) переменными
* **initUI –** основной метод класса, отвечающий за использование функций, создающих или изменяющих окно данного класса
* **view\_database –** метод, осуществляющий отображение таблицы
* **update\_view –** метод, осуществляющий обновление графического представления таблицы
* **clear\_database –** метод, производящий очистку базы данных

**Файл «solution\_for\_cards.py» - back-end**

Класс Table является классом для работы с табличными значениям, созданием таблиц, а также работой с файлами формата «.csv».

**Данный класс также содержит функции (в дальнейшем методы):**

* **\_\_init\_\_ -** метод, отвечающий за инициализацию данного класса, с именованными (имена переменных говорящие) переменными
* **create –** метод, отвечающий за создание таблицы категорий 13х13
* **open\_table –** метод, открывающий файлы формата «.csv», а далее преобразующий их в словарь категорий карт, после чего ещё открывает файл формата «.csv» и преобразует тот уже в словарь цветов категорий, а после передающий оба этих словаря обратно в экземпляр данного класса, с помощью которого был использован данный метод
* **open\_chances\_colors –** метод, открывающий файлы формата «.csv», а далее преобразующий их в словарь цветов шансов под каждый из диапазонов в процентах, передающий их обратно в экземпляр данного класса, с помощью которого был использован данный метод
* **save\_designations –** метод, используемый для сохранения категорий карт в таблицу формата «.csv»
* **save\_colors\_category –** метод, используемый для сохранения цветов категорий карт в таблицу формата «.csv»

**Константы файла:**

* **COMBINATIONS –** список наименований всех комбинаций в покере
* **DIGNITIES –** список наименований всех карт в покере
* **SYMBOLS –** список наименований всех символов карт в покере
* **DIGNITIES\_FOR\_TABLE –** матрица 13х13, сохраняющая в себе таблицу всех вариаций карт игрока, учитывая также одинаковую/разную масти или одинаковые карты
* **CATEGORIES –** список наименований всех категорий
* **DIGNITIES\_SHORT\_TO\_LONG –** словарь, позволяющий изменить короткое наименование достоинства карты в привычное
* **CARDS –** список, хранящий 52 игровых карты

Класс CheckCombinations является классом для проверки различных комбинаций на различных стадиях игры, возвращающий конкретные шансы на победу, измеряемые в процентах.

**Данный класс также содержит функции (в дальнейшем методы):**

* **\_\_init\_\_ -** метод, отвечающий за инициализацию данного класса, с именованными (имена переменных говорящие) переменными
* **check\_preflop –** метод, отвечающий за проверку шансов на префлопе против остальных комбинаций, а после возвращающий конкретный шанс
* **chance\_on\_flop –** метод, отвечающий за проверку шансов на флопе, возвращающий в результате работы словарь в формате:  
  «dictionary[key] = [count\_win, count\_all]»
* **chance\_on\_turn –** метод, отвечающий за проверку шансов на тёрне, возвращающий в результате работы словарь в формате:  
  «dictionary[key] = [count\_win, count\_all]»
* **chance\_on\_river –** метод, отвечающий за проверку шансов на ривере, возвращающий в результате работы словарь в формате:  
  «dictionary[key] = [count\_win, count\_all]»
* **check\_who\_win –** метод, возвращающий 0 или 1 в зависимости от того, комбинация пользователя выигрышная получилась или же нет при расчёте шансов
* **check\_equal\_combinations –** метод, проверяющий старшинство одинаковых по наименованию комбинаций
* **check\_equal\_straight\_flush –** метод, проверяющий старшинство комбинации «стрит-флеш»
* **check\_equal\_four\_of\_a\_kind –** метод, проверяющий старшинство комбинации «каре»
* **check\_equal\_full\_house –** метод, проверяющий старшинство комбинации «фулл-хаус»
* **check\_equal\_flush -** метод, проверяющий старшинство комбинации «флеш»
* **check\_equal\_straight -** метод, проверяющий старшинство комбинации «стрит»
* **check\_equal\_three\_of\_a\_kind –** метод, проверяющий старшинство комбинации «тройка» или же «сет»
* **check\_equal\_two\_pair –** метод, проверяющий старшинство комбинации «две пары»
* **check\_equal\_pair –** метод, проверяющий старшинство комбинации «пара»
* **check\_equal\_high\_card –** метод, проверяющий старшинство «старшей карты»
* **check\_combination –** метод, производящий проверку комбинаций у выбранных карт
* **check\_royal\_flush –** метод, производящий проверку комбинации «флеш рояль»
* **check\_straight\_flush -** метод, производящий проверку комбинации «стрит флеш»
* **check\_four\_of\_a\_kind -** метод, производящий проверку комбинации «каре»
* **check\_full\_house -** метод, производящий проверку комбинации «фулл-хаус»
* **check\_flush -** метод, производящий проверку комбинации «флеш»
* **check\_straight –** метод, производящий проверку комбинации «стрит»
* **check\_three\_of\_a\_kind -** метод, производящий проверку комбинации «тройка» или «сет»
* **check\_pair -** метод, производящий проверку комбинации «пара», в том числе и отслеживающий вариант, что могут быть две пары

Класс Database является классом для работы с базой данных, которая используется в дальнейшем в файле “ui.py” (front-end).

**Данный класс также содержит функции (в дальнейшем методы):**

* **\_\_init\_\_ -** метод, отвечающий за инициализацию данного класса, с именованными (имена переменных говорящие) переменными
* **add\_our\_cards –** метод, добавляющий карты игрока в соответствующую таблицу базы данных
* **add\_cards\_on\_desk –** метод, добавляющий карты на столе во время конкретного круга раздачи – флопа
* **update\_cards\_on\_desk\_on\_turn –** метод, обновляющий колонку карты на конкретном кругу раздачи – тёрне, а также обновляющий и все остальные карты, если те в ходе использования программы изменили, но при этом не сбросили
* **update\_cards\_on\_desk\_on\_river -** метод, обновляющий колонку карты на конкретном кругу раздачи – ривере, а также обновляющий и все остальные карты, если те в ходе использования программы изменили, но при этом не сбросили
* **update\_main\_table\_on\_turn –** обновление главной таблицы, конкретной колонки последней игры в ходе перехода на круг раздачи – тёрн, с обновлением имеющихся карт, а также шанса
* **update\_main\_table\_on\_river -** обновление главной таблицы, конкретной колонки последней игры в ходе перехода на круг раздачи – ривере, с обновлением имеющихся карт, а также шанса
* **add\_execute\_for\_main\_table –** добавление в главную таблицу записи на кругах раздачи – префлопе и флопе. Запись на флопе в дальнейшем обновляется, если же потребуется
* **clear\_all\_tables –** очищение всех таблиц базы данных, что используется в графическом интерфейсе при отображении главной таблицы базы данных

**Место и сроки выполнения работы**

*Сроки работы*

Работа выполнялась с ноября 2021 по февраль 2022.

*Место работы*

На ИТ Полигоне в Государственном бюджетном общеобразовательном учреждении города Москвы «Образовательный центр «Протон».

# **Результаты**

1. Проанализированы существующие решения использования различных приложений в игре в «Покер», у которых основной целью выступало решение логической задачи ввиду расчётов конкретных шансов на разных стадиях данной игры.
2. Разработана программа с графическим интерфейсом, взаимодействующая с табличными значениями и базами данных. Данное решение этой задачи также взаимодействует с самим пользователем и предоставляет наиболее точную информацию на различных стадиях игры.
3. Проведена оценка работоспособности программы с помощью использования реальных примеров игры в «Покер».

# **Описание завершённого продукта**

«Покерный аналитик» – программа, созданная в целях предоставить пользователю возможность получить точные шансы при всех возможных вариациях на каждой из раздач в игре «Покер», а именно в его конкретной разновидности – «Техасском Холдеме».

Ссылка на видео-демонстрацию проектной работы – <https://youtu.be/QF11_zRDZXo>

# Ссылка на репозиторий с проектом – <https://github.com/JaydenPears/Poker-Analyst>

**Список использованной литературы**

1. Давлетханов, Р. Н. Математические методы как основа стратегии игры в покер / Р. Н. Давлетханов. – Текст: непосредственный // Экономика, управление, финансы: материалы III Междунар. науч. конф. (г. Пермь, февраль 2014 г.). – Пермь: Меркурий, 2014. – С.16-17. [Электронный ресурс] URL: https://moluch.ru/conf/econ/archive/93/4819/ (дата обращения: 22.02.2022).
2. К. Ю. Поляков, Е. А. Еремин. Информатика. Углублённый уровень. Учебник для 10 класса в 2 частях. М.: БИНОМ. Лаборатория знаний, 2014.
3. М. Лутц. Изучаем Python. СПб.: Символ-Плюс, 2011.
4. Задачи по программированию. Под ред. С. М. Окулова, М.: БИНОМ. Лаборатория знаний, 2006.
5. С. М. Окулов. Основы программирования. М.: Бином. Лаборатория знаний, 2012.
6. Лутц. Изучаем Python. СПб.: Символ-Плюс, 2011.
7. Информатика и ИКТ. Задачник-практикум в 2 частях. Под ред. И. Г. Семакина и Е. К. Хеннера. М.: БИНОМ. Лаборатория знаний, 2014.
8. Материалы и презентации к урокам в LMS Яндекс.Лицея.
9. Сайт pythonworld.ru — «Python 3 для начинающих».
10. https://www.youtube.com/playlist?list=PLJOzdkh8T5kpIBTG9mM2wVBjh5OpdwBl — Лекции А.В. Умнова, прочитанные в Школе Анализа Данных Яндекса.
11. Сайт официальной документации модуля PyQT для языка программирования Python — https://doc.qt.io/qtforpython/