**Игра-Тетрис «Block Breakers» на языке программирования python с использованием pygame**

**Парахин Александр и Соколович Никита**

* **Идея проекта:**
  + Создание классической игры "Тетрис" с использованием библиотеки Pygame.
  + Цель: разработка игрового приложения для развлечения и изучения основ программирования на Python и библиотеки pygame.
* **Задачи проекта:**
  + Реализация игровой логики (падение фигур, управление, подсчет очков).
  + Создание интуитивно понятного интерфейса.
  + Добавление дополнительных функций (пауза, правила игры, уровни сложности).

**Основные функции**

1. **main()**
   * Точка входа в программу.
   * Инициализирует Pygame, настраивает экран и шрифты.
   * Запускает игровой цикл и отображает начальный экран.
2. **playTetris()**
   * Основной игровой цикл.
   * Управляет логикой игры: движение фигур, проверка столкновений, подсчет очков, отрисовка игрового поля и фигур.
   * Обрабатывает события (нажатия клавиш, пауза, завершение игры).
3. **showGameRules()**
   * Отображает правила игры на экране.
   * Ожидает нажатия клавиши для возвращения в игру.
4. **generateNewFigure()**
   * Генерирует новую фигуру случайной формы и цвета.
   * Возвращает словарь с параметрами фигуры (форма, поворот, координаты, цвет).
5. **addFigureToCup(cup, fig)**

Добавляет фигуру в игровое поле (cup) на основе её текущих координат и формы.

**6.clearFullLines(cup)**

* + Проверяет, заполнены ли строки в игровом поле.
  + Удаляет заполненные строки и сдвигает верхние строки вниз.
  + Возвращает количество удаленных строк для подсчета очков.

**7.drawCup(cup)**

* + Отрисовывает игровое поле (cup) на экране.
  + Включает отрисовку границ и блоков.

**8.renderFigure(fig, pixel\_x=None, pixel\_y=None)**

* + Отрисовывает текущую фигуру на экране.
  + Если указаны pixel\_x и pixel\_y, отрисовывает фигуру в указанных координатах

**9.renderNextFigure(fig)**

* + Отображает следующую фигуру в боковой панели.

**10.showInitialText(text)**

* + Отображает начальный экран с текстом (например, название игры или сообщение о паузе).
  + Ожидает нажатия клавиши для продолжения.

**Вспомогательные функции**

1. **createTextObject(text, font, color)**
   * Создает объект текста для отрисовки на экране.
   * Возвращает поверхность (surface) и прямоугольник (rect) для позиционирования.
2. **render3DText(text, font, color, x, y)**
   * Отрисовывает текст с эффектом 3D (тень).
   * Используется для отображения заголовков и других текстовых элементов.
3. **handleQuit()**
   * Обрабатывает событие выхода из игры (нажатие Esc или закрытие окна).
4. **checkForKeyPress()**
   * Проверяет, была ли нажата какая-либо клавиша.
   * Используется для ожидания ввода пользователя.
5. **isPositionValid(cup, fig, adjX=0, adjY=0)**
   * Проверяет, может ли фигура находиться в текущей позиции или в позиции с учетом смещения (adjX, adjY).
   * Используется для проверки столкновений и корректности перемещения.
6. **isInCup(x, y)**
   * Проверяет, находятся ли координаты (x, y) в пределах игрового поля.
7. **isLineCompleted(cup, y)**
   * Проверяет, заполнена ли строка y в игровом поле.

**8.convertBlockCoords(block\_x, block\_y)**

* + Преобразует координаты блока в игровом поле в пиксельные координаты на экране.

**9.drawBlockOnScreen(block\_x, block\_y, color, pixel\_x=None, pixel\_y=None)**

* + Отрисовывает один блок на экране.
  + Если указаны pixel\_x и pixel\_y, отрисовывает блок в указанных координатах.

**10.calculateSpeed(points)**

* + Вычисляет уровень игры и скорость падения фигур на основе набранных очков.

**11.displayPauseScreen()**

* + Отображает экран паузы.

**12.drawBackground()**

* + Отрисовывает фон игрового экрана (случайные блоки).

**13.terminateGame()**

* + Завершает игру и закрывает окно Pygame.
* **Геймплей:**
  + Управление фигурами с помощью клавиатуры (стрелки, пробел, Enter).
  + Динамическое увеличение сложности с ростом уровня.
  + Подсчет очков и отображение следующей фигуры.
* **Визуальные элементы:**
  + Цветовая палитра для фигур и фона.
  + Эффекты 3D для текста и блоков.
  + Анимация падения фигур.
* **Дополнительные функции:**
  + Пауза и возобновление игры.
  + Адаптивный интерфейс
  + Отображение правил игры.
  + Возможность мгновенного падения фигуры.
* **Выводы по работе:**
  + Проект успешно реализован, все основные функции работают корректно.
  + Игра обладает интуитивно понятным интерфейсом и увлекательным геймплеем.
  + Проект демонстрирует возможности Python и Pygame для создания 2D-игр.
* **Возможности для доработки:**
  + Добавление новых типов фигур и уровней сложности.
  + Реализация мультиплеера (игра на двоих).
  + Интеграция системы рекордов и онлайн-лидерборда.
  + Улучшение графики (анимации, эффекты, звуки).

Пояснение

**Пояснительная записка к проекту "Block Breakers"**

**Введение**

В данной пояснительной записке представлена информация о проекте "Block Breakers", который реализует классическую игру "Тетрис" на языке программирования Python с использованием библиотеки Pygame. Проект разработан студентами Парахиным Александром и Соколовичем Никитой с целью создания развлекательного игрового приложения и изучения основ программирования в контексте разработки игр.

**Цели и задачи проекта**

**Цель проекта**

Основной целью проекта является создание интуитивно понятной и увлекательной игры "Тетрис", которая позволит пользователям не только развлекаться, но и изучать основы программирования на Python и работу с библиотекой Pygame.

**Задачи проекта**

Для достижения поставленной цели были определены следующие задачи:

1. Реализация игровой логики:
   * Падение фигур.
   * Управление фигурами.
   * Подсчет очков.
2. Создание интуитивно понятного интерфейса для пользователя.
3. Добавление дополнительных функций, таких как:
   * Пауза.
   * Правила игры.
   * Уровни сложности.

**Описание функций**

**Основные функции**

1. **main()**:
   * Точка входа в программу.
   * Инициализирует Pygame, настраивает экран и шрифты.
   * Запускает игровой цикл и отображает начальный экран.
2. **playTetris()**:
   * Основной игровой цикл.
   * Управляет логикой игры: движение фигур, проверка столкновений, подсчет очков, отрисовка игрового поля и фигур.
   * Обрабатывает события (нажатия клавиш, пауза, завершение игры).
3. **showGameRules()**:
   * Отображает правила игры на экране.
   * Ожидает нажатия клавиши для возвращения в игру.
4. **generateNewFigure()**:
   * Генерирует новую фигуру случайной формы и цвета.
   * Возвращает словарь с параметрами фигуры (форма, поворот, координаты, цвет).
5. **addFigureToCup(cup, fig)**:
   * Добавляет фигуру в игровое поле (cup) на основе её текущих координат и формы.
6. **clearFullLines(cup)**:
   * Проверяет, заполнены ли строки в игровом поле.
   * Удаляет заполненные строки и сдвигает верхние строки вниз.
   * Возвращает количество удаленных строк для подсчета очков.
7. **drawCup(cup)**:
   * Отрисовывает игровое поле (cup) на экране.
   * Включает отрисовку границ и блоков.
8. **renderFigure(fig, pixel\_x=None, pixel\_y=None)**:
   * Отрисовывает текущую фигуру на экране.
   * Если указаны pixel\_x и pixel\_y, отрисовывает фигуру в указанных координатах.
9. **renderNextFigure(fig)**:
   * Отображает следующую фигуру в боковой панели.
10. **showInitialText(text)**:
    * Отображает начальный экран с текстом (например, название игры или сообщение о паузе).
    * Ожидает нажатия клавиши для продолжения.

**Вспомогательные функции**

1. **createTextObject(text, font, color)**:
   * Создает объект текста для отрисовки на экране.
   * Возвращает поверхность (surface) и прямоугольник (rect) для позиционирования.
2. **render3DText(text, font, color, x, y)**:
   * Отрисовывает текст с эффектом 3D (тень).
   * Используется для отображения заголовков и других текстовых элементов.
3. **handleQuit()**:
   * Обрабатывает событие выхода из игры (нажатие Esc или закрытие окна).
4. **checkForKeyPress()**:
   * Проверяет, была ли нажата какая-либо клавиша.
   * Используется для ожидания ввода пользователя.
5. **isPositionValid(cup, fig, adjX=0, adjY=0)**:
   * Проверяет, может ли фигура находиться в текущей позиции или в позиции с учетом смещения (adjX, adjY).
6. **isInCup(x, y)**:
   * Проверяет, находятся ли координаты (x, y) в пределах игрового поля.
7. **isLineCompleted(cup, y)**:
   * Проверяет, заполнена ли строка y в игровом поле.
8. **convertBlockCoords(block\_x, block\_y)**:
   * Преобразует координаты блока в игровом поле в пиксельные координаты на экране.
9. **drawBlockOnScreen(block\_x, block\_y, color, pixel\_x=None, pixel\_y=None)**:
   * Отрисовывает один блок на экране.
10. **calculateSpeed(points)**:
    * Вычисляет уровень игры и скорость падения фигур на основе набранных очков.
11. **displayPauseScreen()**:
    * Отображает экран паузы.
12. **drawBackground()**:
    * Отрисовывает фон игрового экрана (случайные блоки).
13. **terminateGame()**:
    * Завершает игру и закрывает окно Pygame.

**Геймплей**

Игра управляется с помощью клавиатуры, где игрок может перемещать фигуры, вращать их и ускорять падение. Динамическое увеличение сложности происходит с ростом уровня, что делает игру более увлекательной. Подсчет очков осуществляется на основе удаленных линий, и игроку отображается следующая фигура для планирования ходов.

**Визуальные элементы**

Визуальные элементы включают:

1. **Фигуры**:
   * Фигуры в игре имеют различные формы и цвета, что помогает игрокам легко различать их. Каждая фигура может быть представлена в виде набора блоков, которые могут иметь разные текстуры или границы.
   * Использование 3D-эффектов при отрисовке фигур добавляет глубину и делает игру более динамичной.
2. **Анимация**:
   * Эффекты, такие как исчезновение заполненных линий, могут быть дополнены анимацией, что делает игровой процесс более увлекательным.
3. **Текст**:
   * Все текстовые элементы, такие как названия, подсказки и очки, оформлены с использованием шрифтов, которые хорошо читаются и соответствуют общей стилистике игры.
   * Эффекты 3D для текста, такие как тени, добавляют визуальной привлекательности и делают интерфейс более интересным.

**Интерфейс пользователя**

Интерфейс игры разработан с учетом удобства и интуитивности. Включает в себя:

1. **Игровое поле**:
   * Основное игровое поле четко выделено, с границами, чтобы игроки могли легко увидеть, где находятся фигуры и где они могут падать.
2. **Панели**:
   * Боковая панель может содержать информацию о текущей игре, такую как количество очков, уровень сложности и следующая фигура. Это помогает игрокам планировать свои действия.
3. **Кнопки управления**:
   * Управление игрой осуществляется с помощью клавиатуры, и игроки могут видеть подсказки о том, какие клавиши использовать для управления фигурами.
4. **Меню**:
   * Игра включает меню для начала новой игры, просмотра правил и выхода из игры. Это меню интуитивно понятно и легко доступно.

**Дополнительные функции**

**Пауза и возобновление игры**

* Игра включает возможность паузы, что позволяет игрокам сделать перерыв, не теряя прогресс. При активации паузы на экране отображается соответствующий текст, и игра останавливается до тех пор, пока игрок не решит продолжить.

**Правила игры**

* В проекте предусмотрено отображение правил игры, что помогает новым игрокам быстро ознакомиться с основами. Правила могут быть представлены в виде текста на экране, доступного через меню.

**Мгновенное падение фигур**

* Возможность мгновенного падения фигур позволяет игрокам ускорять процесс, что может быть полезно в напряженных ситуациях, когда необходимо быстро освободить пространство на игровом поле.

**Выводы**

Проект "Block Breakers" успешно реализован, и все основные функции работают корректно. Игра обладает интуитивно понятным интерфейсом и увлекательным геймплеем, что позволяет пользователям наслаждаться процессом. Реализация игры демонстрирует возможности языка Python и библиотеки Pygame для создания 2D-игр.

**Возможности для доработки**

Несмотря на успешную реализацию, проект имеет потенциал для дальнейшего развития. Возможные направления доработки включают:

1. **Добавление новых типов фигур и уровней сложности**:
   * Введение уникальных фигур с особыми свойствами или механиками может разнообразить игровой процесс. Уровни сложности могут включать разные скорости падения фигур и новые правила.
2. **Реализация мультиплеера**:
   * Возможность играть с друзьями или другими игроками онлайн может значительно увеличить интерес к игре.
3. **Интеграция системы рекордов и онлайн-лидерборда**:
   * Введение системы рекордов позволит игрокам соперничать друг с другом, что добавит элемент соревнования и мотивации.
4. **Улучшение графики**:
   * Работы над графикой могут включать создание более сложных анимаций, улучшение текстур и добавление звуковых эффектов, что сделает игру более захватывающей.

**Заключение**

Данная пояснительная записка призвана предоставить полное представление о проекте "Block Breakers", его целях, задачах и возможностях для дальнейшего развития. Проект не только демонстрирует технические навыки разработчиков, но и предлагает увлекательный опыт для пользователей. Если у вас есть дополнительные вопросы или нужно больше информации о конкретных аспектах проекта, пожалуйста, дайте знать!