**МИНИСТЕРСТВО ОБРАЗОВАНИЯ КРАСНОЯРСКОГО КРАЯ**

**КРАЕВОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«КРАСНОЯРСКИЙ КОЛЛЕДЖ ОТРАСЛЕВЫХ ТЕХНОЛОГИЙ И ПРЕДПРИНИМАТЕЛЬСТВА»**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**

По программированию на Python.

Преподаватель: \_\_\_\_\_\_\_ \_\_\_\_\_\_\_С.Ю.Глухов

подпись

дата

Исполнитель

студент: 1 курса ИС24-02п группы\_\_\_\_\_\_\_ \_\_\_\_\_\_Д.А. Карапетян

подпись

дата

**Красноярск, 2025 г.**

**Отчет о выполнении**

**Описание выполненной работы:**

Создана базовая игра “Змейка” на Python с использованием библиотеки Pygame.

**Реализованы:**

* Движение змейки
* Поедание еды и рост змейки
* Проверка столкновений со стенами и с собой
* Простое управление с клавиатуры

Трудности и решения:

* Первоначальная сложность с пониманием логики Pygame (особенно с обработкой событий и отрисовкой). Решение: изучение документации, просмотр примеров кода, эксперименты.
* Сложность с организацией кода (особенно с разделением на классы и функции). Решение: рефакторинг кода, следование принципам объектно-ориентированного программирования.

**План дальнейших действий по изучению программирования на Python и разработке программных проектов:**

* Углубленное изучение Pygame:
* Работа со спрайтами, анимацией, звуком.
* Создание более сложных игровых механик (разные типы еды, бонусы, враги).
* Разработка системы меню и уровней.

**Изучение других библиотек Python:**

* Tkinter или Kivy для создания графических интерфейсов.
* Requests для работы с API.
* Pandas и NumPy для анализа данных.

**Практика на реальных проектах:**

* Разработка небольших приложений для автоматизации задач.
* Участие в open-source проектах.
* Создание веб-приложений с использованием Flask или Django.

**Изучение алгоритмов и структур данных:**

* Понимание основных алгоритмов сортировки, поиска и т.д.
* Использование различных структур данных (списки, словари, множества, деревья) для решения задач.

**Предложить к реализации:**

**Модификация игры “Змейка”:**

* Добавить различные типы еды (бонусы, замедлители).
* Реализовать систему уровней со сменой фона и скорости змейки.
* Добавить возможность игры для двух игроков.
* Создать систему сохранения результатов.

**1 Какие особенности библиотеки Pygame вы изучили в ходе работы над проектом?**

**Инициализация Pygame (pygame.init()):** Необходима для начала работы с библиотекой.

**Создание окна (pygame.display.set\_mode()):** Создает игровое окно заданного размера.

**Управление событиями (pygame.event.get()):** Позволяет отслеживать действия пользователя (нажатия клавиш, движения мыши, закрытие окна).

**Рисование примитивов (pygame.draw.rect()):** Используется для рисования прямоугольников, которые представляют змейку и еду.

**Обновление экрана (pygame.display.flip()):** Отображает все изменения, сделанные на экране.

**Управление частотой кадров (pygame.time.Clock().tick()):** Контролирует скорость игры, задавая максимальное количество кадров в секунду.

**Работа со шрифтами (не используется в этом базовом примере, но важная часть Pygame):** позволяет добавлять текст на экран (счет, сообщения и т.д.).

**Работа со спрайтами (не используется в этом базовом примере, но мощный инструмент):** позволяет создавать более сложные игры с анимацией и управлением столкновениями.

**2 Что собой представляет “Проект” как структура в файловой системе на диске?**

Проект — это организованный набор файлов и папок, которые вместе составляют программу или приложение.

**В файловой системе проекта обычно есть:**

* **Исходный код:** Python файлы (.py) с инструкциями программы;
* **Ресурсы:** Изображения, звуки, шрифты и другие медиафайлы;
* **Файлы конфигурации:** Файлы с настройками проекта;
* **Файлы документации:** Описания проекта, инструкции по использованию;
* **Файлы виртуального окружения (если используется):** Папка, содержащая изолированный набор установленных пакетов.

**3 Что такое виртуальное окружение? Как и когда его нужно создавать?**

Виртуальное окружение — это изолированное пространство для Python проектов. Оно позволяет устанавливать пакеты и зависимости для конкретного проекта, не влияя на другие проекты или глобальную установку Python.

**Как создать виртуальное окружение:**

* **Установите virtualenv (если еще не установлен):** pip install virtualenv;
* **Создайте виртуальное окружение в папке проекта:** virtualenv venv (где venv - имя папки для окружения, можно выбрать другое).

**Активируйте виртуальное окружение:**

* Windows: venv\Scripts\activate;
* macOS/Linux: source venv/bin/activate.

**Когда создавать:**

* Для каждого нового Python проекта;
* Когда проект требует определенные версии пакетов, которые могут конфликтовать с другими проектами;
* Чтобы обеспечить воспроизводимость проекта на других машинах.

**4 В чем особенности использования выбранной вами среды разработки при работе над созданием игрового проекта?**

**VS Code:**

* **Интеграция с Python:** Автоматическое определение Python интерпретатора, поддержка автодополнения кода, отладка;
* **Расширения:** Наличие расширений для Pygame (например, для подсветки синтаксиса, отладки);
* **Интегрированный терминал:** удобно для запуска и управления виртуальным окружением;
* **Удобная работа с файлами проекта:** легко создавать, перемещать и редактировать файлы;
* **Git интеграция:** Удобно для контроля версий.

**5 Выделите приоритеты структуры программного кода. Раскрыть линейную последовательность с примерами, в каком порядке, что за чем следует при создании игрового проекта на pygame?**

При создании проекта на Pygame, структура кода (порядок действий) часто имеет следующий приоритет и линейную последовательность:

* **Импорт библиотек:** import pygame, random (и других необходимых);
* **Инициализация Pygame:** pygame.init();
* **Определение констант и переменных:** Размеры экрана, цвета, скорость змейки и т.д. (часто в settings.py);
* **Создание окна игры**: screen = pygame.display.set\_mode((SCREEN\_WIDTH, SCREEN\_HEIGHT));
* Создание игровых объектов: Создание экземпляров классов змейки, еды и т.д. snake = Snake(), food = Food().

**Основной цикл игры (while running:):**

* **Обработка событий:** Отслеживание нажатий клавиш, закрытия окна;
* **Обновление игровых объектов:** Перемещение змейки, проверка столкновений;
* **Рисование**: Очистка экрана, рисование змейки, еды;
* **Обновление экрана**: pygame.display.flip();
* **Управление частотой кадров:** clock.tick(SNAKE\_SPEED);
* **Выход из Pygame:** pygame.quit().

**6 Зачем в проекте используются: циклы, условия, данные, структуры данных, функции? Выделить и пояснить их назначение с особенностью реализации в проекте? Что можно, или нужно изменить?**

**Циклы (while, for):**

* **Назначение:** Повторение блоков кода. В основном цикле игры while running: происходит обновление и отрисовка каждого кадра. Цикл for в snake.draw() перебирает сегменты змейки для отрисовки;
* **Особенности:** управляют ходом игры, обеспечивают непрерывное обновление состояния и отображения;
* **Что изменить:** можно добавить разные игровые режимы, которые будут иметь свои циклы.

**Условия (if, elif, else):**

* **Назначение:** Выполнение различных блоков кода в зависимости от условий. Проверка столкновений змейки, обработка нажатий клавиш для изменения направления движения;
* **Особенности:** Определяют логику игры, реакцию на действия пользователя и события в игре;
* **Что изменить:** добавить условия для разных уровней сложности, бонусов, Game Over.

**Данные (числа, строки, булевы значения):**

* **Назначение:** Хранение информации об игре. Позиция змейки, размеры экрана, цвета, счет;
* **Особенности:** представляют состояние игры, используются для вычислений и принятия решений;
* **Что изменить:** добавить данные о здоровье змейки, таймеры, бонусы.

**Структуры данных (списки, кортежи):**

* **Назначение:** Организация и хранение данных. Список snake.body хранит координаты всех сегментов змейки. Кортежи используются для представления координат и направлений движения;
* **Особенности:** обеспечивают удобный доступ к данным и возможность их изменения;
* **Что изменить:** использовать словари для более сложной организации данных (например, для хранения информации о бонусах).

**Функции (def):**

* **Назначение:** Разделение кода на логические блоки, упрощение повторного использования кода. Функции для отрисовки, движения змейки, проверки столкновений;
* **Особенности:** улучшают читаемость и поддерживаемость кода;
* **Что изменить:** добавить функции для обработки меню, загрузки уровней, отображения результатов.

**ЗАКЛЮЧЕНИЕ**

Таким образом мы научились работать с библиотекой Pygame, сделав игру змейка, но это ещё не полное изучение библиотеки и её возможностей в дальнейшем будем углубляться и добавлять графику и остальную составляющую для создание более сложных игр.