## **Техническое задание**

**Проект**: Игра «Змейка» на Python с использованием библиотеки pygame.

**Цель**: Реализация классической игры "Змейка" с базовыми механиками: движение змеи, появление пищи, рост змеи, проверка на столкновения, подсчет очков.

### **Общее описание:**

Игра "Змейка" - классическая аркадная игра, где игрок управляет змеей, которая растет, поедая еду. Цель игры - набрать как можно больше очков, избегая столкновений со стенами или самой собой.

### **Функциональные требования:**

#### **Основной игровой процесс:**

* **Управление змеей:** Игрок может управлять змеей с помощью клавиш клавиатуры (например, стрелок) для движения вверх, вниз, влево и вправо.
* **Рост змеи:** Змея увеличивается в длину при поедании пищи.
* **Появление пищи:** Пища появляется в случайном месте на игровом поле.
* **Окончание игры:** Игра заканчивается, если змея столкнется со стеной или самой собой.

#### **Визуальное оформление:**

* **Игровое поле:** Простое прямоугольное поле с четкими границами.
* **Змея:** Визуальное представление змеи, состоящее из последовательности блоков.
* **Пища:** Ярко выделенный объект, отличающийся от фона и змеи.
* **Интерфейс:** Отображение текущего счета и возможно других элементов интерфейса (например, таймера, уровня сложности).

### **Основные алгоритмы:**

1. **Движение змеи:**
   * Храним координаты каждого сегмента змеи в списке или массиве.
   * При каждом шаге:
     + Сдвигаем все сегменты на одну позицию назад.
     + Добавляем новую голову змеи в начало списка.
     + Удаляем последний элемент списка, если змея не съела пищу.
2. **Появление пищи:**
   * Генерируем случайные координаты для пищи, проверяя, чтобы она не совпадала с текущим положением змеи или другими сегментами пищи.
3. **Проверка столкновений:**
   * Проверяем, не вышла ли голова змеи за границы игрового поля.
   * Проверяем, не пересекается ли голова змеи с ее телом (проходим по всему списку сегментов).
4. **Увеличение змеи:**
   * При поедании пищи не удаляем последний элемент списка, таким образом увеличивая длину змеи.

### **Структуры данных:**

* **Список (или массив):** Используется для хранения координат каждого сегмента змеи. Каждый элемент списка представляет собой кортеж (x, y) или объект с двумя полями (x, y).
* **Переменные:**
  + x1, y1: Координаты головы змеи.
  + food\_x, food\_: Координаты пищи.
  + zmey\_list: Список координат всех сегментов змеи.
  + zmeydlin: Длина змеи.
  + game\_over: Флаг, указывающий на окончание игры.

### **Основной цикл игры:**

1. **Обработка событий:** Проверяем, нажаты ли клавиши управления.
2. **Обновление состояния:**
   * Обновляем координаты головы змеи в зависимости от нажатых клавиш.
   * Проверяем столкновения.
   * Если змея съела пищу, увеличиваем ее длину и генерируем новую пищу.
   * Обновляем список координат змеи.
3. **Отрисовка:**
   * Очищаем экран.
   * Рисуем змею, пищу и счет.
   * Обновляем экран.

### **Тестирование:**

* **Функциональное тестирование:** Проверка корректности игрового процесса, управления змеей, роста, появления пищи, окончания игры.
* **Визуальное тестирование:** Проверка правильности отображения игровых элементов, шрифтов, цветов и других визуальных аспектов.