

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ  
ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных систем

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Дата сдачи на проверку:

«\_\_\_» 2025 г.

Проверено:

«\_\_\_» 2025 г.

Отчёт по лабораторной работе №1  
по дисциплине  
«Теория Автоматов»

Разработал студент гр. ИВТб-2301-05-00

\_\_\_\_\_ /Черкасов А. А./

(подпись)

Старший Преподователь

\_\_\_\_\_ /Мельцов В. Ю./

(подпись)

Работа защищена

«\_\_\_» 2025 г.

Киров

2025

## Цель лабораторной работы

Разработка автомата, подающего случайную последовательность входных сигналов.

## Задание

Разработать автомат, подающий случайную последовательность входных сигналов. Автомат моделирует процесс управления пирамидами с кольцами, где каждая пирамида может находиться в одном из трех состояний:

1. **Начальное состояние (O)**: массив пар (индекс пирамиды, количество колец), задающий начальную конфигурацию.
2. **Состояние обработки (P)**: массив пирамид, находящихся в процессе обработки. Изначально копирует состояние O.
3. **Завершенное состояние (F)**: массив пирамид, достигших максимального количества колец (12).

В каждом шаге симуляции автомат выполняет следующие действия:

- Случайным образом выбирает пирамиду из состояния P и добавляет ей одно кольцо.
- Если количество колец достигает 12, пирамида перемещается в состояние F.
- С заданной вероятностью (параметр emergencyChance) вызывает аварийную ситуацию: выбирает случайную пирамиду из P с количеством колец не менее 3 и уменьшает ее кольца на 3.

Создать приложение с графическим интерфейсом на Qt6, позволяющее:

1. Настраивать начальное состояние пирамид (количество колец).
2. Выбирать цвета для визуализации пирамид.
3. Управлять скоростью симуляции и вероятностью аварий.
4. Визуализировать текущее состояние пирамид с анимацией.
5. Сохранять и загружать конфигурации симуляции.

## Реализация приложения

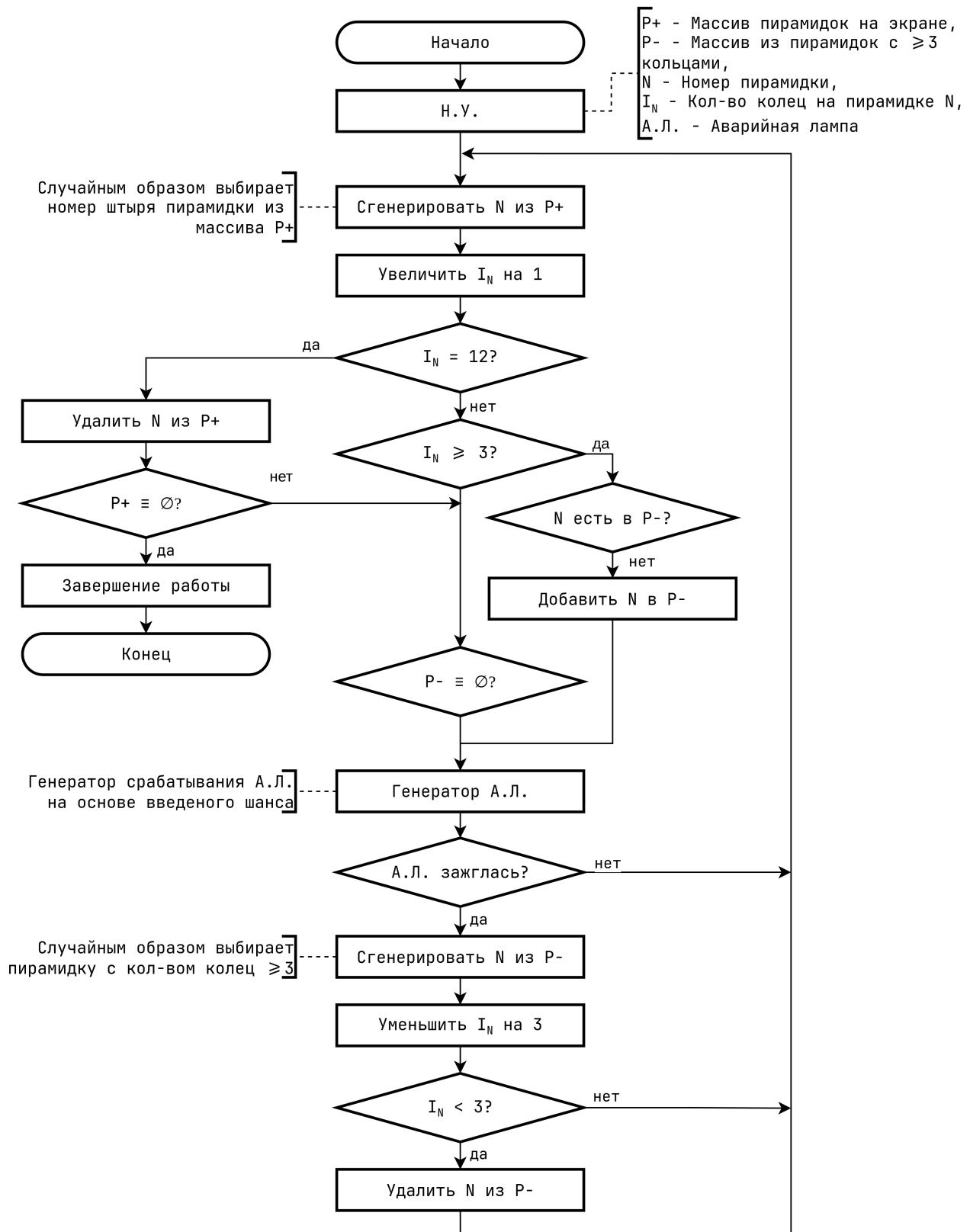


Рисунок 1 - Схема алгоритма автомата

Архитектура приложения построена на фреймворке Qt6 и включает следующие основные классы:

- `Simulation` — класс, реализующий логику автомата с состояниями O, P, F
- `MainWindow` — главное окно приложения с пользовательским интерфейсом
- `PyramidWidget` — виджет для визуализации пирамид с кольцами и анимацией
- `InitialFillDialog` — диалог настройки начального количества колец
- `ColorDialog` — диалог выбора цветов пирамид

## Интерфейс программы

Главное окно приложения состоит из следующих элементов управления:

- **Строка меню** (вверху): содержит меню «Файл» (Открыть, Сохранить, Выход), «Настройки» (Начальное заполнение, Выбор цвета) и «Справка» (О программе, Об авторе).
- **Область визуализации пирамид** (центр): 10 виджетов пирамид, расположенных в 2 ряда по 5 штук. Каждая пирамида отображается своим цветом с кольцами.
- **Панель управления** (внизу): содержит элементы управления симуляцией.
- **Строка состояния** (внизу): показывает количество пирамид в состояниях P и F.

Панель управления включает:

- **Ползунок «Скорость (тиков/сек)»** (0-100): регулирует скорость симуляции. Значение 0 останавливает симуляцию.
- **Поле ввода скорости:** текстовое поле для точного ввода значения скорости с валидацией (только цифры 0-100).
- **Ползунок «Шанс аварии (%)»** (0-100): задает вероятность аварийной ситуации в каждом тике.
- **Поле ввода шанса аварии:** текстовое поле для ввода процента аварий с валидацией.
- **Кнопка «Старт/Стоп»:** запускает или останавливает симуляцию. При запус-

ке текст меняется на «Стоп».

- **Кнопка «Пауза/Продолжить»:** приостанавливает или возобновляет симуляцию. При паузе текст меняется на «Продолжить».
- **Кнопка «Выход»:** закрывает приложение.

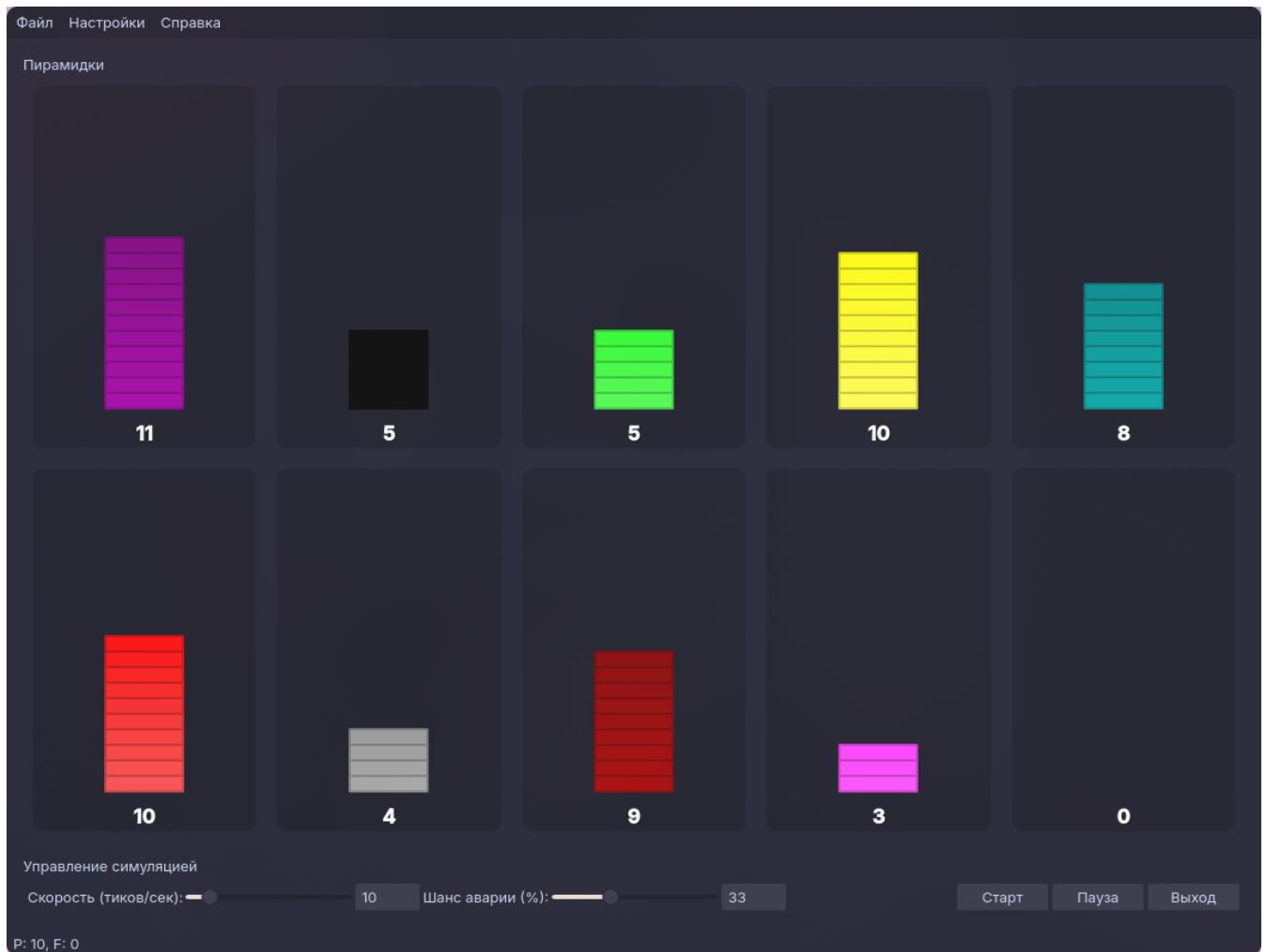


Рисунок 2 - Главное окно приложения с визуализацией пирамид

Диалог настройки начального количества колец открывается через меню «Настройки → Начальное заполнение». Содержит:

- **10 полей ввода (QSpinBox)** для каждой пирамиды с диапазоном 0-11 колец.
- **Кнопка «Случайно»:** заполняет все поля случайными значениями от 0 до 11.
- **Кнопка «Сброс»:** устанавливает 0 колец для всех пирамид.

- **Кнопка «OK»:** применяет настройки и закрывает диалог.
- **Кнопка «Отмена»:** закрывает диалог без сохранения изменений.

Логика работы: при нажатии «Случайно» генерируются случайные значения для имитации различных начальных условий. «Сброс» позволяет быстро установить нулевые значения.

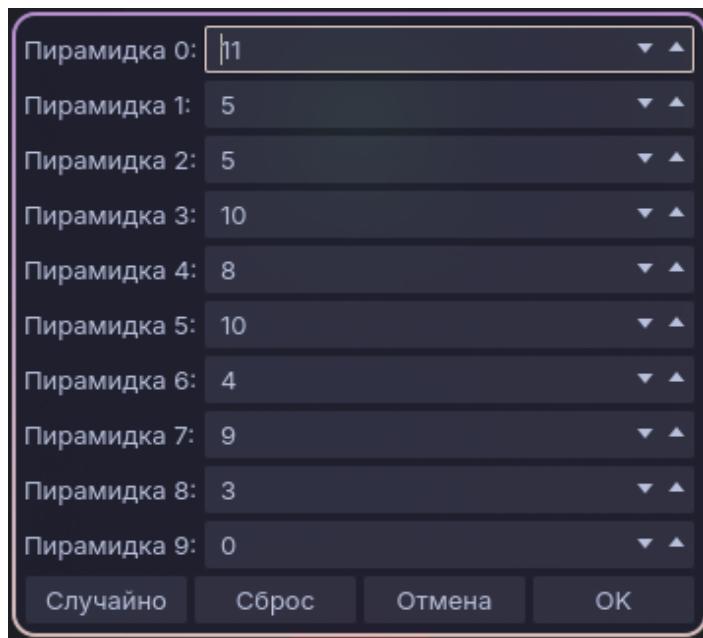


Рисунок 3 - Диалог настройки начального количества колец

Диалог выбора цветов открывается через меню «Настройки → Выбор цвета». Содержит:

- **10 выпадающих списков** (QComboBox) для каждой пирамиды с 14 цветами палитры.
- **Цветные иконки** рядом с названиями цветов для визуального выбора.
- **Кнопка «Случайно»:** присваивает каждой пирамиде уникальный случайный цвет.
- **Кнопка «OK»:** применяет настройки цветов.
- **Кнопка «Отмена»:** закрывает диалог без изменений.

Логика работы: при выборе цвета в одном списке система проверяет, не используется ли этот цвет другой пирамидой. Если конфликт обнаружен, выводится

предупреждение и автоматически выбирается доступный цвет. Кнопка «Случайно» гарантирует уникальность цветов путем случайного перераспределения.

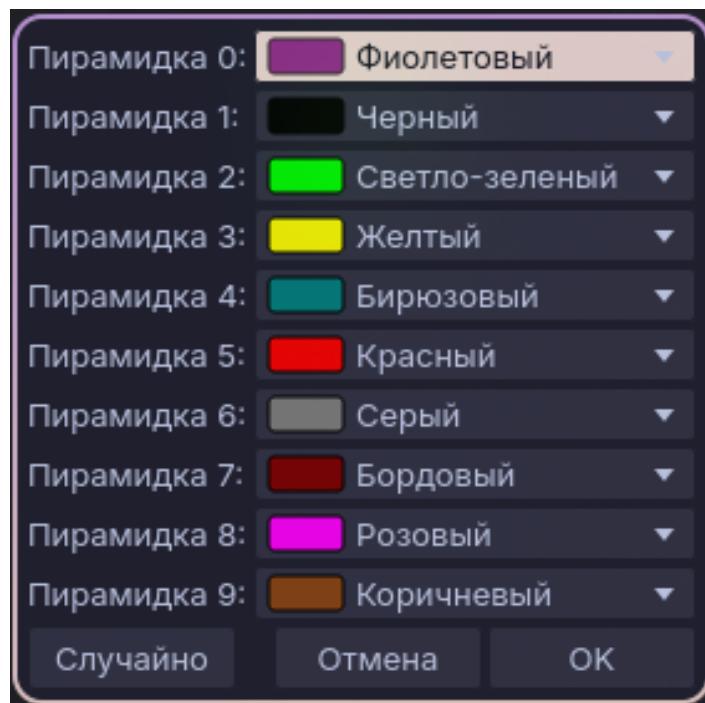


Рисунок 4 - Диалог выбора цветов пирамид

Во время работы симуляции пирамиды анимируются: кольца появляются постепенно, при авариях происходит визуальный эффект тревоги. Статусная строка показывает текущее распределение пирамид по состояниям.

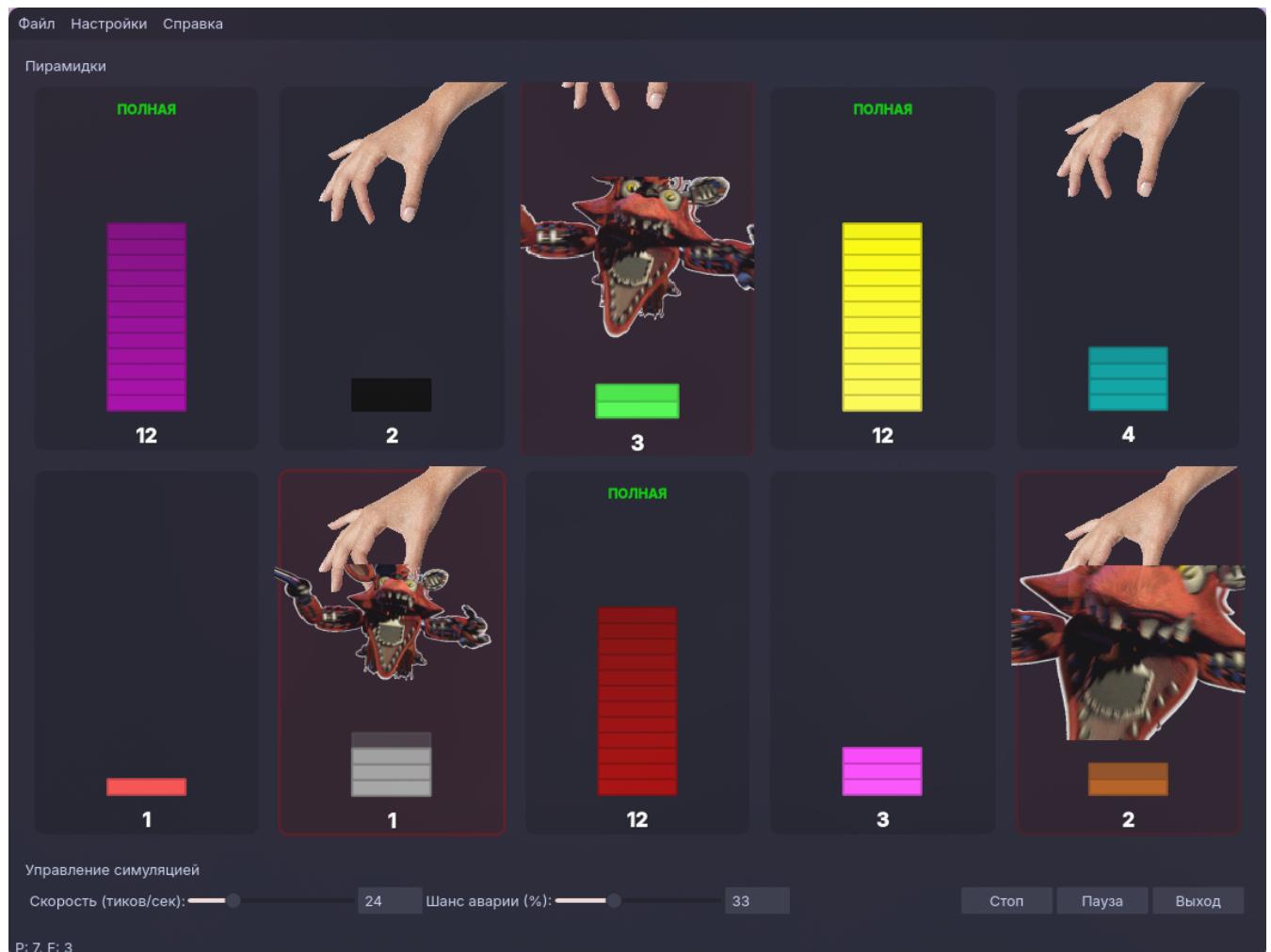


Рисунок 5 - Процесс симуляции с анимацией

## Выход

В ходе выполнения лабораторной работы №1 был разработан конечный автомат, моделирующий процесс управления пирамидами с кольцами. Автомат реализует три состояния (O, P, F) и генерирует случайную последовательность входных сигналов в виде добавления колец и аварийных ситуаций.

Приложение создано на фреймворке Qt6 с использованием языка C++. Реализована визуализация состояния автомата с анимацией, диалоги настройки парамет-

ров, сохранение и загрузка конфигураций. Код организован в модули с четким разделением ответственности между классами `Simulation`, `MainWindow`, `PyramidWidget` и диалогами.