МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«КУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет физики, математики, информатики

Кафедра информационной безопасности

ЛАБОРАТОРНАЯ РАБОТА 2

по дисциплине

«Интерфейсы программирования приложений»

На тему: «Проектирование пользовательского интерфейса десктопного приложения»

Обучающейся 2 курса очной формы

обучения

направления подготовки

09.03.01 Информатика и вычислительная техника

Направленность (профиль)

Прикладной искусственный интеллект

Жаанбаев Баймурат Кенешбекович

Руководитель:

старший преподаватель кафедры ПОАИС

Ураева Елена Евгеньевна

Курск, 2025

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc199940668)

[Теоретическая часть 4](#_Toc199940669)

[1.1. Используемые технологии 4](#_Toc199940670)

[1.2. Основные элементы интерфейса 4](#_Toc199940671)

[Практическая часть 6](#_Toc199940672)

[2.1. Разработка интерфейса в Qt Designer 6](#_Toc199940673)

[2.2. Программная реализация 7](#_Toc199940674)

[2.3. Демонстрация работы программы 8](#_Toc199940675)

[2.3.1. Скриншоты 8](#_Toc199940676)

[2.3.2. Тестирование программы 9](#_Toc199940677)

[Анализ результатов 9](#_Toc199940678)

[ЗАКЛЮЧЕНИЕ 11](#_Toc199940679)

[ПРИЛОЖЕНИЕ А 12](#_Toc199940680)

[ПРИЛОЖЕНИЕ Б 14](#_Toc199940681)

[ПРИЛОЖЕНИЕ В 19](#_Toc199940682)

# ВВЕДЕНИЕ

В рамках данной лабораторной работы рассматривается процесс разработки графического интерфейса пользователя (GUI) для приложения-генератора случайных чисел с использованием языка программирования Python и библиотеки PyQt6. Целью работы является изучение основ проектирования интерфейсов с помощью инструмента Qt Designer, освоение механизмов работы с элементами управления, такими как кнопки и таблицы, а также реализация функционала генерации случайных чисел и их отображения в табличном виде.

Разработка GUI-приложений является важной частью современного программирования, поскольку удобный и интуитивно понятный интерфейс повышает эффективность взаимодействия пользователя с программой. Использование PyQt6 и Qt Designer позволяет упростить процесс создания графических приложений, обеспечивая гибкость и визуальный подход к проектированию. Данная работа направлена на практическое применение полученных теоретических знаний, что способствует закреплению навыков разработки программного обеспечения.

Отчет включает теоретическую часть, описывающую используемые технологии и элементы интерфейса, практическую часть с процессом разработки и демонстрацией работы программы, а также анализ результатов и выводы. Структура отчета позволяет последовательно раскрыть этапы выполнения задания и оценить достигнутые результаты.

# Теоретическая часть

## 1.1. Используемые технологии

Для разработки графического интерфейса приложения-генератора случайных чисел были применены следующие технологии:

1. PyQt6 – современная библиотека Python для создания графических интерфейсов. Она предоставляет набор классов и инструментов для разработки кроссплатформенных приложений с богатым функционалом. PyQt6 является портом фреймворка Qt, обеспечивая доступ к виджетам, обработке событий и управлению интерфейсом.
2. Qt Designer – визуальный инструмент для проектирования графических интерфейсов. Позволяет создавать макеты окон, размещать элементы управления (виджеты) и настраивать их свойства без необходимости написания кода. Результат сохраняется в файл формата .ui, который затем интегрируется в Python-программу.
3. Модель сигналов и слотов – ключевой механизм PyQt6 для обработки взаимодействия между элементами интерфейса. Сигналы (например, нажатие кнопки) связываются со слотами (функциями), что позволяет реагировать на действия пользователя.
4. Модуль random – встроенный модуль Python, предназначенный для генерации псевдослучайных чисел. В данной работе используется функция random.randint() для создания целых чисел в заданном диапазоне.

Эти технологии обеспечивают удобство разработки, гибкость настройки и надежность работы приложения.

## 1.2. Основные элементы интерфейса

В процессе разработки были использованы следующие ключевые элементы интерфейса из библиотеки PyQt6:

1. QMainWindow – класс, представляющий главное окно приложения. Служит основой для размещения виджетов, меню и панелей инструментов. Обеспечивает базовую структуру окна с возможностью настройки заголовка, размеров и положения.
2. QPushButton – виджет кнопки, предназначенный для инициирования действий пользователя. В данном приложении кнопка используется для запуска процесса генерации случайных чисел.
3. QTableWidget – виджет таблицы, позволяющий отображать данные в виде сетки из строк и столбцов. Поддерживает настройку размеров, добавление элементов и автоматическую подстройку колонок.
4. QTableWidgetItem – класс для создания элементов таблицы. Каждый элемент представляет значение в ячейке и позволяет задавать текст, выравнивание и другие свойства.

Эти компоненты обеспечивают базовую функциональность приложения, позволяя пользователю взаимодействовать с программой и получать сгенерированные данные в удобном виде.

# Практическая часть

## 2.1. Разработка интерфейса в Qt Designer

Разработка графического интерфейса началась с создания макета в Qt Designer. Был создан файл generator.ui, определяющий структуру и внешний вид приложения. Основные элементы интерфейса включают:

1. Главное окно (QMainWindow): основа приложения, содержащая все остальные виджеты (см. рис. 1). Свойства окна:

* Заголовок: "Генератор случайных чисел".
* Размеры: 600x400 пикселей (начальные размеры, которые можно изменить).

1. Кнопка (QPushButton): элемент для запуска генерации чисел (см. рис.1).

* Имя объекта: gener.
* Текст: "Сгенерировать".
* Размещение: в правой части окна, выравнивание по центру.

1. Таблица (QTableWidget): элемент для отображения сгенерированных чисел (см. рис. 2).

* Имя объекта: tableWidget.
* Начальные размеры: 10 строк и 3 столбца (задаются программно).
* Свойства: включена автоматическая подстройка размеров колонок под содержимое.

В Qt Designer виджеты были размещены следующим образом: кнопка "Сгенерировать" расположена в правой части окна, а таблица занимает основное пространство в левой части. Свойства, такие как размеры, шрифты и выравнивание, были настроены для обеспечения удобства и читаемости. После завершения проектирования файл generator.ui был сохранен для дальнейшей интеграции с кодом.

## 2.2. Программная реализация

Для реализации функциональности приложения был написан код на Python с использованием PyQt6.

Основные этапы:

1. Установка и настройка PyQt6:

Библиотека PyQt6 была установлена с помощью команды pip install PyQt6 в виртуальной среде Python. Это обеспечило доступ к необходимым классам и модулям.

1. Подключение интерфейса к коду:

Файл generator.ui был подключен к программе с помощью функции uic.loadUi. Это позволило загрузить макет интерфейса непосредственно в объект класса MainWindow.

1. Описание класса MainWindow:

Создан класс MainWindow, наследуемый от QMainWindow. В методе \_\_init\_\_ выполняется:

* Загрузка интерфейса из файла generator.ui.
* Подключение сигнала нажатия кнопки gener к слоту (методу) generate\_sequence с помощью self.gener.clicked.connect(self.generate\_sequence).
* Настройка политики автоматического изменения размеров таблицы с использованием setSizeAdjustPolicy(QAbstractItemView.SizeAdjustPolicy.AdjustToContents).

1. Логика генерации случайных чисел:

Метод generate\_sequence реализует основную функциональность:

* Очищает таблицу от предыдущих данных с помощью self.tableWidget.clear().
* Устанавливает фиксированное количество строк (10) и столбцов (3) с помощью методов setRowCount и setColumnCount.
* Генерирует случайные целые числа в диапазоне от 1 до 100 с использованием random.randint(1, 100).
* Заполняет таблицу сгенерированными значениями, преобразуя их в строки и добавляя в ячейки через QTableWidgetItem.

1. Подстройка таблицы:

Для улучшения читаемости колонок таблицы применяется метод resizeColumnsToContents(), который автоматически подстраивает ширину колонок под содержимое.

**Примечание -** Код программы приведен в приложении А и Б.

## 2.3. Демонстрация работы программы

Программа была протестирована для проверки корректности работы (см. рис. 2). Основные шаги взаимодействия пользователя:

1. Пользователь запускает приложение, открывается главное окно с заголовком "Генератор случайных чисел".
2. В окне отображается кнопка "Сгенерировать" и пустая таблица.
3. Пользователь нажимает кнопку "Сгенерировать".
4. Программа генерирует случайные числа и заполняет таблицу (10 строк, 3 столбца) значениями в диапазоне от 1 до 100.
5. Таблица автоматически подстраивает ширину колонок для удобного отображения.

### 2.3.1. Скриншоты

1. Окно при запуске: показывает пустую таблицу и кнопку "Сгенерировать" (см. Приложение В, рисунок В.1).
2. Результат генерации чисел: демонстрирует таблицу, заполненную случайными числами (см. Приложение В, рисунок В.1).

Скриншоты работы программы приведены в приложении В. Приложение успешно выполняет задачу генерации и отображения данных, обеспечивая простое и понятное взаимодействие.

### 2.3.2. Тестирование программы

Для проверки работоспособности приложения было проведено тестирование:

1. Проверка корректности генерации чисел:

* Многократное нажатие кнопки "Сгенерировать" подтвердило, что числа генерируются случайным образом в диапазоне от 1 до 100 (см. Приложение В, рисунок В.3).
* Значения в ячейках таблицы различны при каждом запуске, что соответствует принципу случайности (см. Приложение В, рисунок В.4).

1. Проверка отображения данных (см. Приложение В, рисунок В.3):

* Таблица корректно отображает 10 строк и 3 столбца.
* Ширина колонок автоматически подстраивается под содержимое, обеспечивая читаемость.

Тестирование показало, что программа работает стабильно и соответствует базовым требованиям.

# Анализ результатов

Достигнутые цели:

1. Создан работающий графический интерфейс с использованием PyQt6 и Qt Designer.
2. Реализована функциональность генерации случайных чисел и их отображения в таблице QTableWidget.
3. Интерфейс прост и интуитивно понятен, пользователь может легко запустить генерацию данных.

Ограничения текущей версии:

1. Фиксированный размер таблицы (10 строк, 3 столбца) не позволяет пользователю изменять количество строк и столбцов.
2. Диапазон чисел (1–100) задан жестко в коде, что ограничивает гибкость приложения.
3. Отсутствует возможность сохранения сгенерированных данных.

Возможные улучшения:

1. Добавление настроек: включить поля ввода (например, QLineEdit или QSpinBox) для задания диапазона чисел (минимальное и максимальное значения) и размеров таблицы (строки и столбцы).
2. Экспорт данных: реализовать функцию сохранения таблицы в файл, например, в формате CSV, для дальнейшего использования.
3. Визуальное оформление: добавить стили (цвета фона, шрифты, границы таблицы) для улучшения внешнего вида.
4. Обработка ошибок: внедрить проверку ввода пользователем (например, некорректных чисел) с выводом сообщений об ошибках через QMessageBox.

Эти улучшения сделают приложение более универсальным и удобным для различных задач, таких как тестирование, анализ данных или обучение.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы был успешно разработан графический интерфейс приложения-генератора случайных чисел с использованием Python и библиотеки PyQt6. Изучены основы работы с инструментом Qt Designer, применены элементы управления, такие как кнопки (QPushButton) и таблицы (QTableWidget), а также реализован функционал генерации случайных чисел и их отображения в таблице.

Практическая часть работы позволила освоить процесс проектирования интерфейса в Qt Designer, подключения его к программному коду и реализации логики генерации данных. Программа успешно выполняет поставленные задачи, обеспечивая генерацию и визуализацию случайных чисел, что подтверждается скриншотами и описанием взаимодействия пользователя с интерфейсом.

Анализ результатов выявил возможности для улучшения, такие как добавление настроек диапазона чисел, изменение размеров таблицы и экспорт данных в файл. Освоенные навыки работы с PyQt6 и Qt Designer создают основу для дальнейшего развития проекта, включая повышение удобства использования и расширение функциональности. Таким образом, лабораторная работа способствовала закреплению теоретических знаний и приобретению практического опыта в области разработки GUI-приложений.

# ПРИЛОЖЕНИЕ А

Листинг А.1 - Листинг кода (generator.py)

import sys  
from PyQt6.QtWidgets import (  
 QApplication,  
 QMainWindow,  
 QTableWidgetItem,  
 QAbstractItemView  
)  
from PyQt6 import uic  
import random  
  
  
class MainWindow(QMainWindow):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 # Загружаем UI файл, созданный в Qt Designer  
 uic.loadUi('generater.ui', self)  
 # Замените 'your\_form.ui' на имя вашего файла  
  
 # Предполагаем, что у вас есть кнопка с именем "generateButton"  
 # и таблица "tableWidget"  
  
 # Эти имена должны соответствовать именам объектов в Qt Designer  
 self.gener.clicked.connect(self.generate\_sequence)  
  
 # Настраиваем политику автоматического изменения размеров  
 self.tableWidget.setSizeAdjustPolicy(QAbstractItemView.SizeAdjustPolicy.AdjustToContents)  
  
 def generate\_sequence(self):  
 # Очистка предыдущего содержимого таблицы  
 self.tableWidget.clear()  
  
 # Установка количества строк и столбцов  
 rows = 10 # Можно изменить количество строк  
 columns = 3 # Можно изменить количество столбцов  
 self.tableWidget.setRowCount(rows)  
 self.tableWidget.setColumnCount(columns)  
  
 # Генерация и заполнение таблицы случайными числами  
 for i in range(rows):  
 for j in range(columns):  
 # Генерируем случайное число (можно изменить диапазон)  
 random\_value = random.randint(1, 100)  
 # Создаем элемент таблицы и устанавливаем значение  
 item = QTableWidgetItem(str(random\_value))  
 self.tableWidget.setItem(i, j, item)  
  
 # Автоматическая подстройка размера колонок под содержимое  
 # self.tableWidget.resizeColumnsToContents()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app = QApplication(sys.argv)  
 window = MainWindow()  
 window.show()  
 sys.exit(app.exec())

# ПРИЛОЖЕНИЕ Б

Листинг Б.1 - Файл макета (generator.ui)

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>MainWindow</class>

<widget class="QMainWindow" name="MainWindow">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>486</width>

<height>338</height>

</rect>

</property>

<property name="windowTitle">

<string>MainWindow</string>

</property>

<widget class="QWidget" name="centralwidget">

<layout class="QHBoxLayout" name="horizontalLayout\_3">

<item>

<layout class="QVBoxLayout" name="verticalLayout">

<item>

<spacer name="verticalSpacer\_2">

<property name="orientation">

<enum>Qt::Vertical</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>20</width>

<height>40</height>

</size>

</property>

</spacer>

</item>

<item>

<widget class="QLabel" name="label">

<property name="text">

<string>Генератор случайных чисел и строк</string>

</property>

</widget>

</item>

<item>

<spacer name="verticalSpacer\_4">

<property name="orientation">

<enum>Qt::Vertical</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>20</width>

<height>40</height>

</size>

</property>

</spacer>

</item>

<item>

<layout class="QGridLayout" name="gridLayout">

<item row="2" column="1">

<widget class="QLabel" name="label\_2">

<property name="text">

<string>Длина строки</string>

</property>

</widget>

</item>

<item row="2" column="0">

<widget class="QSpinBox" name="spinBox"/>

</item>

<item row="6" column="0" colspan="2">

<widget class="QCheckBox" name="checkBox\_4">

<property name="text">

<string>Специальные символы (!@#)</string>

</property>

</widget>

</item>

<item row="3" column="0" colspan="2">

<widget class="QCheckBox" name="checkBox">

<property name="text">

<string>Буквы верхнего регистра (A-Z)</string>

</property>

</widget>

</item>

<item row="4" column="0" colspan="2">

<widget class="QCheckBox" name="checkBox\_2">

<property name="text">

<string>Буквы нижнего регистра (a-z)</string>

</property>

</widget>

</item>

<item row="5" column="0" colspan="2">

<widget class="QCheckBox" name="checkBox\_3">

<property name="text">

<string>Цифры (0-9)</string>

</property>

</widget>

</item>

</layout>

</item>

<item>

<spacer name="verticalSpacer\_3">

<property name="orientation">

<enum>Qt::Vertical</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>20</width>

<height>40</height>

</size>

</property>

</spacer>

</item>

<item>

<widget class="QPushButton" name="gener">

<property name="text">

<string>Генерировать</string>

</property>

</widget>

</item>

<item>

<spacer name="verticalSpacer">

<property name="orientation">

<enum>Qt::Vertical</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>20</width>

<height>40</height>

</size>

</property>

</spacer>

</item>

<item>

<layout class="QHBoxLayout" name="horizontalLayout">

<item>

<widget class="QPushButton" name="pushButton\_3">

<property name="text">

<string>Настройки</string>

</property>

</widget>

</item>

<item>

<widget class="QPushButton" name="pushButton">

<property name="text">

<string>Справка</string>

</property>

</widget>

</item>

</layout>

</item>

</layout>

</item>

<item>

<widget class="QTableWidget" name="tableWidget"/>

</item>

</layout>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>486</width>

<height>18</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources/>

<connections/>

</ui>

# ПРИЛОЖЕНИЕ В

Скриншоты работы программы

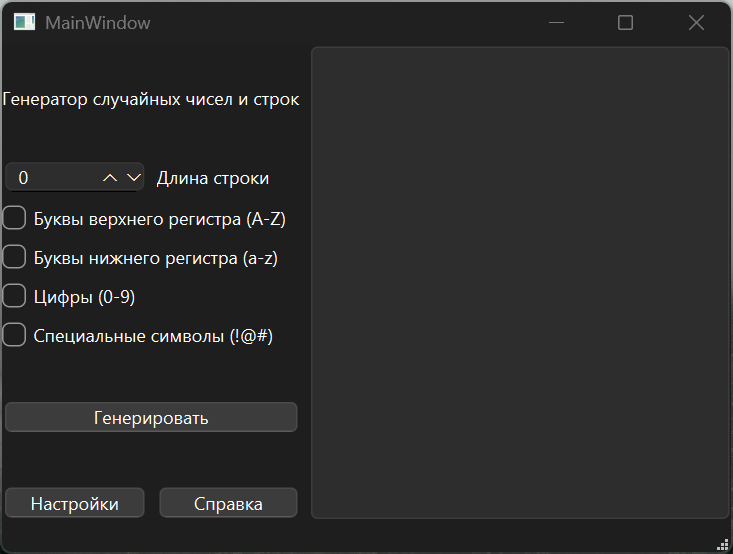


Рисунок В.1 - Главное окно

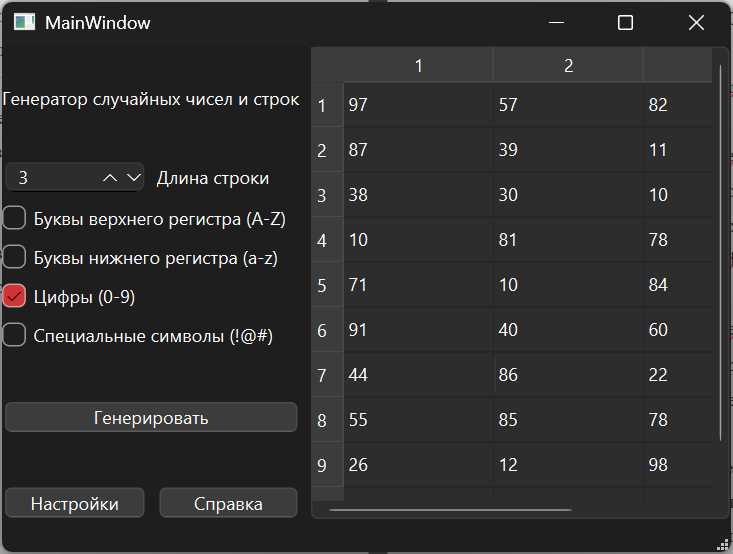


Рисунок В.2 - Отображение сгенерированных чисел

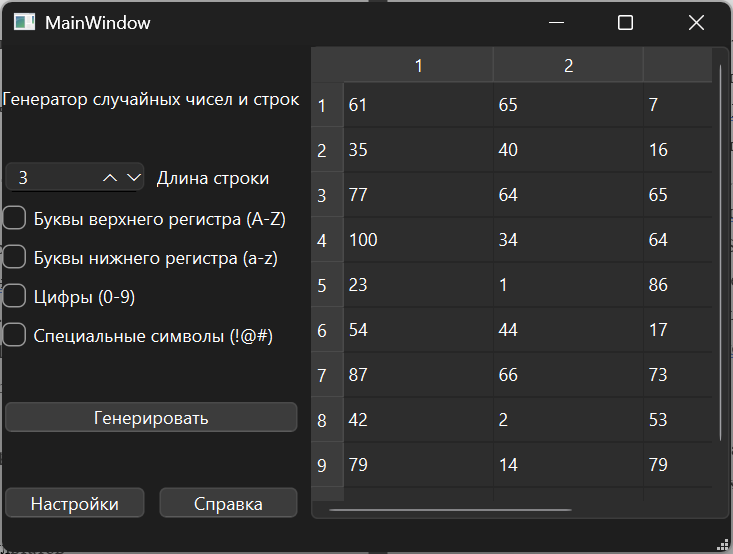


Рисунок В.3 - Тестирование №1

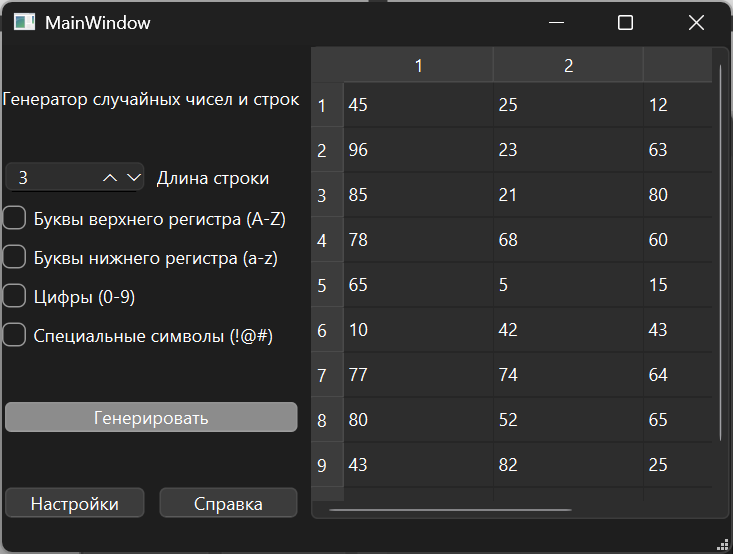


Рисунок В.4 - Тестирование №2