

|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  Федеральное государственное автономное образовательное учреждение высшего образования  **«Дальневосточный федеральный университет»**  **(ДВФУ)** |
| **ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ**  **Департамент математического и компьютерного моделирования** |
| **О Т Ч Е Т**  о прохождении учебной практики.  Технологическая (проектно-технологическая) практика  направление подготовки 02.03.01 «Математика и компьютерные науки»  профиль «Сквозные цифровые технологии» |

|  |  |
| --- | --- |
| Отчет защищена  с оценкой \_\_\_\_\_\_\_\_ | Выполнил студент  группы № Б9121-02.03.01сцт  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г. |
| Рег. № \_\_\_\_\_\_\_\_\_\_\_ | Руководитель практики  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(должность, уч.звание)*  **\_\_\_**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(Ф.И.О.) (подпись)*  «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г. |
| «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022 г. | Практика пройдена в срок  с «18» июля 2022 г.  по «30» июля 2022 г.  (2 недели) |

г. Владивосток

2022

Содержание

[Введение 1](#_Toc110304283)

[Настройка 2](#_Toc110304284)

[Python и объектно-ориентированное программирование (ООП) 2](#_Toc110304285)

[PyQt5 3](#_Toc110304286)

[Pygame 4](#_Toc110304287)

[Другие библиотеки 4](#_Toc110304288)

[Проект А (PyQt5) – ASCII converter 6](#_Toc110304289)

[Project B (Pygame) – Conway’s game of life 9](#_Toc110304290)

[Заключение 10](#_Toc110304291)

[Список литературы 11](#_Toc110304292)

[Приложения 12](#_Toc110304293)

Введение

В течение двух недель группа практиковалась с Python и применением таких вещей, как функции и объектно-ориентированное программирование, чтобы использовать две библиотеки Python для разработки графических интерфейсов и игр. Первая библиотека — PyQt5, а вторая — Pygame. В течение этого периода было выполнено много упражнений, чтобы применить на практике многие функции, содержащиеся в этих библиотеках, и необходимые инструменты, имеющиеся в Python.

Первое приложение использует PyQt5, который преобразует изображения в строки ASCII. Второе приложение использует Pygame и представляет собой визуализатор игры жизни Конвея. Также в первом приложении есть функция экспорта строки ASCII в визуализатор игры Conway. Все это использовало множество функций для разработки графического интерфейса приложений и разработки логики, работающей за кулисами.

Настройка

Для начала необходима установка Python. Для этого проекта 64-разрядный Python 3.10.5 использовался с Visual Studio Code в качестве текстового редактора в Windows 11. Чтобы иметь больший контроль над библиотеками, используемыми в проекте, и упростить создание файла требований, для него была создана виртуальная среда.

Требования к проекту включают один пакет помимо PyQt5 и Pygame. Для управления файлами изображений использовалась библиотека изображений, в данном случае Pillow.

|  |
| --- |
| **requirements.txt** |
| Pillow==9.2.0  pygame==2.1.2  PyQt5==5.15.7 |

# Python и объектно-ориентированное программирование (ООП)

При использовании пакетов будет использоваться ООП, поэтому уместно дать общий обзор того, как Python управляет объектами и их свойствами.

В Python классы можно использовать для создания объектов класса, которые поддерживают ссылки на атрибуты и создание экземпляров. Последнее относится к вызову объекта класса для создания объекта экземпляра. Ссылки на атрибуты - это единственная операция, понятная объектам экземпляра, где единственными допустимыми именами атрибутов являются атрибуты данных и методы. Класс может действовать как родитель другого класса и наследовать его методы и атрибуты дочернему классу. (Python Software Foundation, 2022)

Короче говоря, экземпляр класса может иметь некоторые атрибуты данных (например, число, строку, массив и т.д.) или метод (который представляет собой функцию, определенную внутри класса). Следующий пример определяет класс MyClass и определяет начальный атрибут «data» для каждого экземпляра с помощью метода \_\_init\_\_, а также создает метод f, который возвращает атрибут data, который в данном случае представляет собой список, в обратном порядке порядок. Как видно, создав экземпляр класса в переменной x, можно получить доступ к значениям, хранящимся для этого экземпляра, и использовать определенные для него методы. Это будет полезно в ходе реализации проекта.

|  |
| --- |
| **Пример класса в Python** |
| >>> class MyClass(ParentClass):  ... def \_\_init\_\_(self):  ... self.data = [1,2,3]  ... def f(self):  ... return self.data[::-1]  ...  >>> x = MyClass() # создание экземпляра  >>> x.data # атрибут  [1, 2, 3]  >>> x.f() # метод  [3, 2, 1] |

# PyQt5

PyQt помогает создавать графические пользовательские интерфейсы (GUI), такие как окна, диалоги и т. д. Это также дает возможность добавлять кнопки, способы ввода данных пользователем, просмотр текста и т. д. PyQt в основном использует объектно-ориентированное программирование, используя классы из пакет в качестве родительских классов для класса, созданного программистом, чтобы показать их графический интерфейс.

Для более сложных проектов и более красивого кода мы используем Qt Designer — приложение, помогающее разрабатывать пользовательские интерфейсы и экспортировать их в файлы .ui. Чтобы использовать их на Python, мы можем использовать класс uic на PyQt5, чтобы импортировать дизайн в код и получить доступ к методам, принадлежащим каждому объекту, присутствующему в дизайне. Это также помогает сфокусировать код на реализации функций, которые работают в самой программе.

# Pygame

Pygame — это набор модулей Python, предназначенных для написания видеоигр. Pygame добавляет функциональность поверх превосходной библиотеки SDL. (Pygame, б.д.)

В этой библиотеке есть различные способы рисования мультимедиа и применения логики к экрану для создания игр с помощью Python, и мы увидим, сколько разных применений у этой библиотеки.

# Другие библиотеки

* PIL: Библиотека изображений Python (Python Image Library) добавляет возможности обработки изображений в ваш интерпретатор Python. Эта библиотека обеспечивает обширную поддержку форматов файлов, эффективное внутреннее представление и довольно мощные возможности обработки изображений. (Clark, б.д.)
* os: Чтобы открыть диалоговое окно файла в первой программе, каталог, установленный для диалогового окна, устанавливается с помощью os.getcwd(), который возвращает текущий рабочий каталог.
* sys: sys.argv используется для указания аргументов командной строки, передаваемых сценарию, и используется с PyQt при создании переменной приложения.
* copy: Этот пакет стандартной библиотеки имеет функцию глубокого копирования для копирования значений объектов без ссылок, что будет полезно при работе с двумерными матрицами в Python в проекте B.

Проект А (PyQt5) – ASCII converter

В первом проекте использовались модули os, sys и PyQt5; и класс ConwaysGame из файла Project B. Класс ASCIIConverter наследуется от класса QMainWindow из модуля PyQt5, и это будет класс, управляющий окном приложения.

В верхней части окна есть заголовок с использованием QLabel. Слева есть разделы с кнопками для каждой функции, а справа есть QTextEdit, который покажет пользователю изображение при преобразовании в текст. (См. Приложение 1)

Во-первых, мы можем добавить изображение, которое откроет диалоговое окно файла в текущем каталоге для выбора файла png, jpg или jpeg. После выбора маршрут файла будет сохранен и показан на экране, а также использован для создания экземпляра изображения с библиотекой PIL. При преобразовании изображения в оттенки серого каждый пиксель будет иметь уникальное значение, которое мы назовем интенсивностью. В зависимости от интенсивности и количества символов, выбранных пользователем, он выбирает символ ASCII из атрибута CHARS класса, который представляет собой строку символов, упорядоченную по количеству места, которое они занимают в каждом из своих полей.

Чтобы сопоставить каждый символ со значением интенсивности, программа заменяет матрицу значений интенсивности их соответствующими индексами в отсортированном наборе всех присутствующих интенсивностей. Затем, используя соотношение индекса и длины списка, индекс набора интенсивности может быть отображен как индекс списка подмножества символов.

|  |
| --- |
| **Вспомогательная функция для сопоставления интенсивности пикселей с символом** |
| def map\_pix\_to\_char(pixels: list, char\_subset: list) -> list:  intensities = sorted(list(set(pixels)))  pixel\_intensities = [intensities.index(pixel) for pixel in pixels]  intensity\_span = len(intensities) - 1  char\_span = len(char\_subset) - 1  mapped\_chars = []  for pixel in pixel\_intensities:  ratio = float(pixel) / float(intensity\_span)  mapped\_chars.append(char\_subset[-char\_span + int(ratio \* char\_span)])  return mapped\_chars |

После сопоставления каждого символа с каждым значением интенсивности строка затем форматируется и печатается в поле QTextEdit, а также обрабатывается соответствующее изменение размера окна в зависимости от размера, выбранного пользователем.

В следующем групповом поле пользователь может изменить настройки представления ASCII. С помощью ползунков можно изменить размер (который изменяет ширину и сохраняет постоянное отношение высоты к ширине) или количество символов (что обычно показывает идеальный вывод около среднего значения символов).

Наконец, последний раздел имеет дополнительные функции, такие как возможность скопировать в буфер обмена текущее преобразованное изображение, сохранить текст в файл .txt или активировать «секретную функцию» программы.

Чтобы скопировать текст, он выбирает текущий текст в QTextEdit и копирует его с помощью QTextEdit.copy(). Чтобы сохранить текст в файл, он берет обычный текст с правой стороны, открывает текстовый файл в текущем каталоге и записывает текст в файл. И эта ранее упомянутая секретная функция генерирует преобразование с настройками по умолчанию для размера и количества символов, преобразует его в матрицу и превращает каждый символ в 1 или 0, используя тот же алгоритм, который использовался ранее для сопоставления интенсивности с символами. Эта матрица передается в экземпляр класса ConwaysGame, который будет запускать окно с другим проектом. По сути, он будет использовать матрицу в качестве сетки по умолчанию, на которой пользователь может наблюдать за игрой жизни Конвея. (См. Приложение 2, Приложение 3 и Приложение 4)

Project B (Pygame) – Conway’s game of life

Игра жизни Конвея — это клеточный автомат, созданный математиком Джоном Конвеем в 1970 году (Numberphile, 2014). Это игра с нулевым игроком, в которой эволюция доски определяется ее начальным состоянием. На этой доске есть клетки, которые могут быть мертвыми или живыми. Правила следующие:

|  |
| --- |
| **Правила игры жизни Конвея** |
| 1. Живая клетка будет жить, если вокруг нее будет 2 или 3 соседа. 2. Мертвая клетка оживет, если у нее будет 3 соседа. 3. Все остальные клетки умрут в следующем поколении. |

Эта программа рисует на черном экране сетку плиток размером 30x30, а также имеет две кнопки для приостановки или сброса визуализации, которые включены в папку с ресурсами для их хранения. Для управления этими кнопками использовался класс для хранения того, были они нажаты или нет. Также были добавлены некоторые функциональные возможности для некоторых клавиш клавиатуры и щелчков мышью и указаны в начале игры на терминале. При запуске игра выводит правила на терминал для игрока.

При нажатии ESC игра сбрасывается и отображает исходную матрицу, где все 1 представляют собой прямоугольники, окрашенные в фиолетовый цвет, а все 0 не нарисованы. При вводе ПРОБЕЛа состояние игры изменяется, и программа перестает обновлять следующую итерацию поля в каждом кадре. Затем можно закрасить или стереть ячейки, щелкнув левой и правой кнопкой мыши соответственно. (См. Приложение 3 и Приложение 4)

Заключение

Эти две недели применили на практике мои предыдущие знания и помогли мне научиться использовать новые полезные инструменты в Python для разных целей. Хотя есть некоторые улучшения, которые можно было бы сделать с окончательным состоянием проектов, было очень сложно применить на практике эти недавно приобретенные навыки. Но очень интересно, что удалось сделать два разных проекта и даже внедрить один в другой.

Список литературы

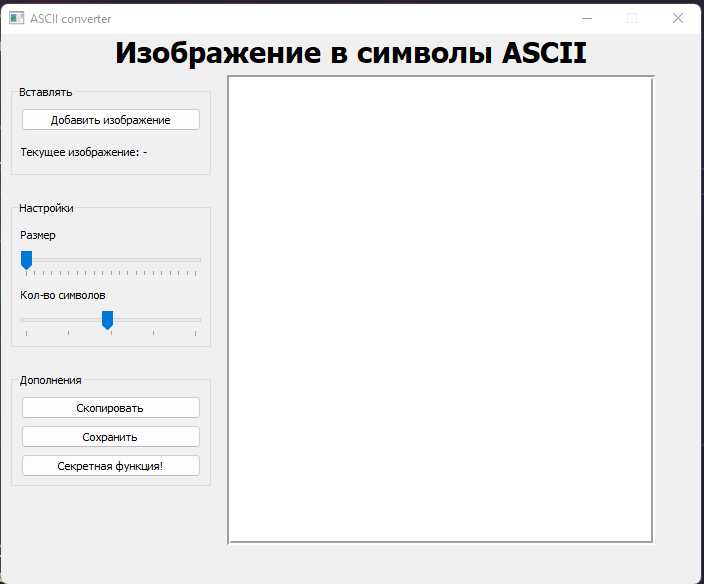
Clark, A. (б.д.). *Overview*. Получено из Pillow (PIL Fork) 9.2.0 documentation: https://pillow.readthedocs.io/en/stable/

Numberphile. (4 03 2014 г.). *Does John Conway hate his Game of Life?* Получено из Youtube: https://www.youtube.com/watch?v=E8kUJL04ELA

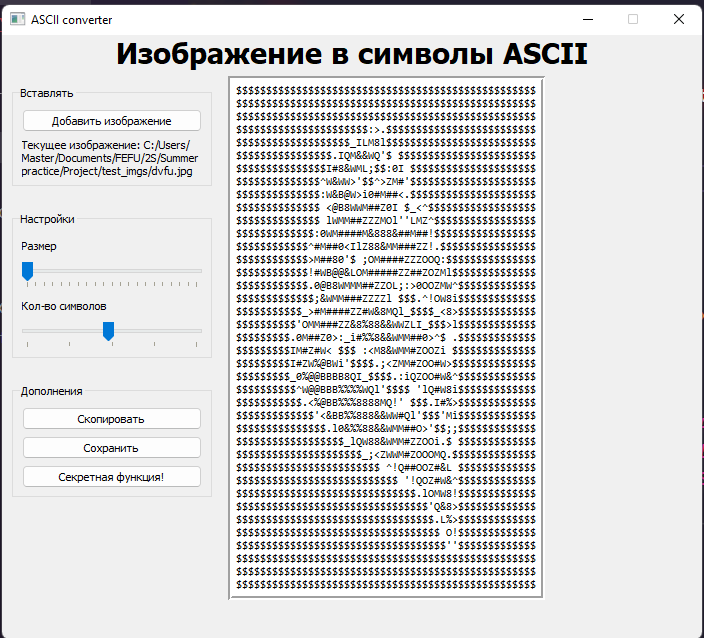
Pygame. (б.д.). *About*. Получено из pygame wiki: https://www.pygame.org/wiki/about

Python Software Foundation. (16 06 2022 г.). *9. Classes*. Получено из Python 3.10.5 documentation: https://docs.python.org/3/tutorial/classes.html

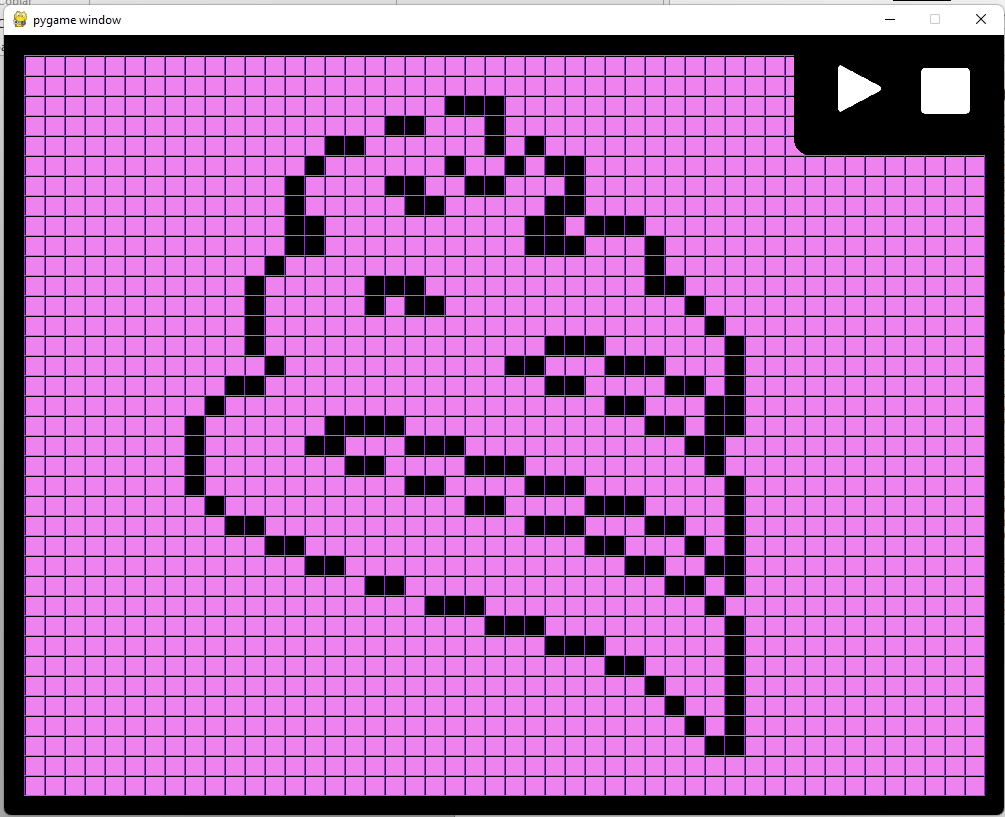
Приложения



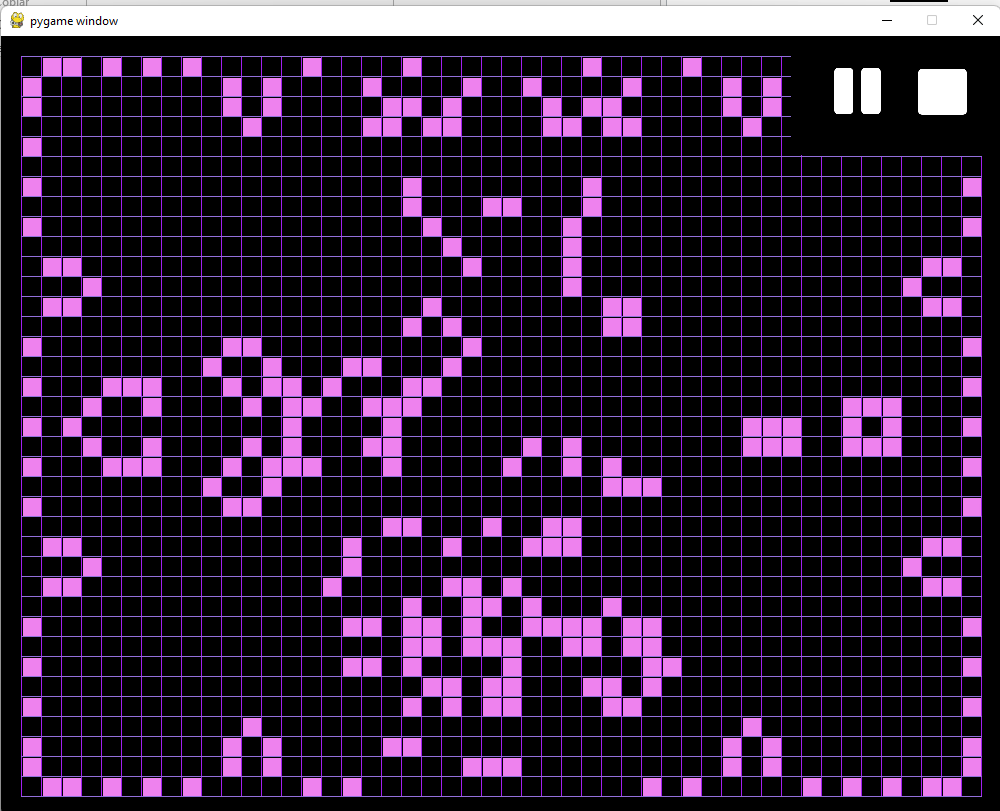
Приложение 1. Главное окно приложения ASCII Converter.



Приложение 2. Тест с использованием логотипа ДВФУ.



Приложение 3. Тест с использованием логотипа ДВФУ (игра жизни Конвея — исходное состояние).



Приложение 4. Тест с использованием логотипа ДВФУ (игра жизни Конвея - бег).