МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Балтийский федеральный университет имени Иммануила Канта»

ОНК «Институт высоких технологий»

ОТЧЁТ О ПРОХОЖДЕНИИ

УЧЕБНОЙ ТЕХНОЛОГИЧЕСКОЙ (ПРОЕКТНО-ТЕХНОЛОГИЧЕСКОЙ) ПРАКТИКИ

на базе Высшей школы компьютерных наук и искусственного интеллекта

Выполнил Строков Данил Викторович

студент очной формы обучения 1 курса

направления подготовки 01.03.02 Прикладная математика и информатика

профиль обучения «Информатика и программирование»

Руководитель практики

Ст. преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Тарачков М.В.

г. Калининград 2024 г.

# ОГЛАВЛЕНИЕ

[**ВВЕДЕНИЕ 3**](#_Toc172103460)

[**ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ ЗНАНИЯ 4**](#_Toc172103461)

[**Язык программирования Python и его особенности 4**](#_Toc172103462)

[**Объектно-ориентированное программирование 5**](#_Toc172103463)

[**Библиотека tkinter в Python 6**](#_Toc172103464)

[**Модуль random в Python 7**](#_Toc172103465)

[**ГЛАВА 2. ЗАДАНИЕ НА ПРАКТИКУ 9**](#_Toc172103466)

[**Задача игра «Snake» 9**](#_Toc172103467)

[**Реализация 9**](#_Toc172103468)

[**ГЛАВА 3. ВЫПОЛНЕНИЕ ЗАДАНИЯ НА ПРАКТИКУ 11**](#_Toc172103469)

[**Пункт 1. 11**](#_Toc172103470)

[**Пункт 2. 12**](#_Toc172103471)

[**Пункт 3. 12**](#_Toc172103472)

[**Пункт 4. 13**](#_Toc172103473)

[**Пункт 5. 14**](#_Toc172103474)

[**Пункт 6. 14**](#_Toc172103475)

[**Пункт 7. 15**](#_Toc172103476)

[**Пункт 8. 16**](#_Toc172103477)

[**Пункт 9. 17**](#_Toc172103478)

[**ЗАКЛЮЧЕНИЕ 18**](#_Toc172103479)

[**СПИСОК ЛИТЕРАТУРЫ 19**](#_Toc172103480)

[**ПРИЛОЖЕНИЕ 1 20**](#_Toc172103481)

# ВВЕДЕНИЕ

Вид практики – Учебная технологическая (проектно-технологическая) практика (далее Учебная практика).

Цель учебной практики: получение первичных профессиональных умений навыков.

Задачи учебной практики:

1. Закрепление и углубление теоретических знаний в области информационных технологий;
2. Закрепление и углубление теоретических знаний в области информационных технологий;

# ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ ЗНАНИЯ

# Язык программирования Python и его особенности

Python — это высокоуровневый язык программирования общего назначения, который широко используется благодаря своей простоте и читаемости. Он был создан Гвидо ван Россумом и впервые выпущен в 1991 году. Python поддерживает несколько парадигм программирования, включая объектно-ориентированное, процедурное и функциональное программирование.

Особенности Python:

* Простота и читаемость кода: Синтаксис Python интуитивно понятен и похож на английский язык, что делает его легким для изучения и использования.
* Интерпретируемость: Python является интерпретируемым языком, что означает, что код выполняется строчка за строчкой, что упрощает тестирование и отладку.
* Большая стандартная библиотека: Python имеет обширную стандартную библиотеку, которая предоставляет множество полезных функций и модулей для выполнения разнообразных задач.
* Кроссплатформенность: Python работает на различных операционных системах, таких как Windows, macOS и Linux.
* Поддержка различных парадигм программирования: Python поддерживает процедурное, объектно-ориентированное и функциональное программирование.

# Объектно-ориентированное программирование

Объектно-ориентированное программирование — это парадигма программирования, основанная на концепции "объектов", которые могут содержать данные и методы для обработки этих данных. В контексте игр, объекты могут представлять игровые сущности, такие как персонажи, враги, предметы и другие элементы.

Основные принципы ООП:

* Инкапсуляция: Объединение данных и методов, работающих с этими данными, в один объект. Это помогает скрыть внутреннюю реализацию и защитить данные от некорректного использования.
* Наследование: Возможность создавать новые классы на основе существующих, наследуя их свойства и методы. Это способствует повторному использованию кода.
* Полиморфизм: Способность использовать объекты разных классов через общий интерфейс, что позволяет создавать гибкие и расширяемые системы.
* Абстракция: Выделение значимых характеристик объекта и игнорирование незначительных деталей. Это упрощает понимание и использование объектов.

Программирование игр относится к объектно-ориентированному программированию (ООП), однако также часто использует другие концепции, такие как управление состоянием, события и асинхронное программирование для создания интерактивных и отзывчивых приложений.

# Библиотека tkinter в Python

**Tkinter** – это пакет для Python, предназначенный для работы с библиотекой Tk. Библиотека Tk содержит компоненты графического интерфейса пользователя (graphical user interface – GUI). Эта библиотека написана на языке программирования Tcl.

Под графическим интерфейсом пользователя (GUI) подразумеваются все те окна, кнопки, текстовые поля для ввода, скроллеры, списки, радиокнопки, флажки и другие элементы, которые можно увидеть на экране, открывая то или иное приложение. Через них происходит взаимодействие с программой и управление ею. Все эти элементы интерфейса называются виджетами.

В настоящее время почти все приложения, которые создаются для конечного пользователя, имеют GUI. Редкие программы, подразумевающие взаимодействие с человеком, остаются консольными.

Существует множество библиотек GUI, среди которых Tk не самый популярный инструмент, хотя с его помощью написано немало проектов. Он был выбран для Python по-умолчанию. Установочный файл интерпретатора Python обычно уже включает пакет tkinter в составе стандартной библиотеки.

Tkinter можно представить как переводчик с языка Python на язык Tcl. Человек пишет программу на Python, а код модуля tkinter переводит его инструкции на язык Tcl, который понимает библиотека Tk.

Основные свойства tkinter:

* Кроссплатформенность: Приложения, созданные с использованием tkinter, работают на Windows, macOS и Linux.
* Простота использования: tkinter имеет простой и интуитивно понятный API, который позволяет быстро создавать графические интерфейсы.
* Гибкость: tkinter поддерживает создание сложных интерфейсов с использованием виджетов и компоновщиков.

Основными виджетами tkinter являются:

* Tk - Основное окно приложения.
* Label - Текстовая метка.
* Button - Кнопка.
* Entry - Текстовое поле для ввода.
* Text - Многострочное текстовое поле.
* Canvas - Холст для рисования.
* Frame - Контейнер для других виджетов.

# Модуль random в Python

random — это стандартный модуль Python для генерации случайных чисел и выполнения операций с использованием случайности. Модуль предоставляет функции для генерации случайных чисел, выбора случайных элементов из последовательностей и перетасовки списков.

Основные функции модуля random:

* random() - Возвращает случайное число с плавающей точкой в диапазоне [0.0, 1.0).
* randint(a, b) - Возвращает случайное целое число N, такое что a<=N<= b.
* choice(seq) - Возвращает случайный элемент из непустой последовательности seq.
* shuffle(seq) - Перемешивает элементы последовательности seq на месте.
* sample(seq, k) - Возвращает список из k уникальных элементов, выбранных из последовательности seq.

Модуль random полезен в игровом программировании для создания случайных событий, генерации уровней, размещения объектов и других задач, связанных с случайностью.

# ГЛАВА 2. ЗАДАНИЕ НА ПРАКТИКУ

# Задача игра «Snake»

Реализовать игру «Snake», в функции которой будет входить:

* Передвижение по полю
* Сбор красных яблок
* Учёт количества собранных яблок в отдельной плашке с названием «Score»
* Завершение игры при столкновении со стенкой или при съедении змейки самой себя

# Реализация

Разобьём задание на пункты для последовательного выполнения:

1. Установить размеры окна игры, добавить и адаптировать плашку Score под размеры окна;
2. Создать класс, описывающий яблоко, которое будет появляться в рандомных местах окна;
3. Создать класс, описывающий змейку (её цвет и количество частей);
4. Создать функцию движения змейки;
5. Задать для змейки направление движения через стрелки на клавиатуре;
6. Задать исключения для передвижения;
7. Реализовать набор очков и удлинение змейки при съедении яблока;
8. Реализовать конец игры при касании стенки окна игры или при съедении змейки самой себя;
9. Сделать сообщение о конце игры на экране.

# ГЛАВА 3. ВЫПОЛНЕНИЕ ЗАДАНИЯ НА ПРАКТИКУ

# Пункт 1.

Первым делом были импортированы функции библиотеки tkinter и модуль random для дальнейшей работы. Затем были заданы значения размеры окна и параметры игры.

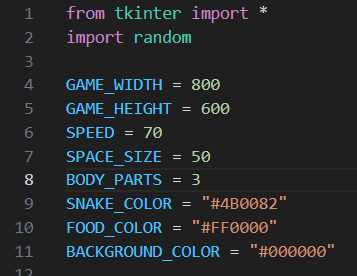


Рисунок 1.

Далее, используя встроенные методы библиотеки tkinter Label(плашка) и Canvas (поле игры), были созданы плашка Score для подсчета съеденных яблок и поле для игры.

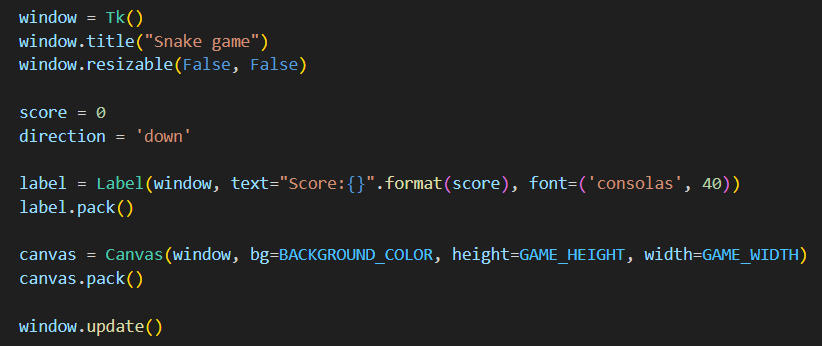


Рисунок 2.

# Пункт 2.

Для определения объекта «яблоко» был создан отдельный класс Food, в котором была реализована система рандомного расположения этого объекта внутри поля игры. Для этого использовался модуль random, импортируемый в п. 1 выше.

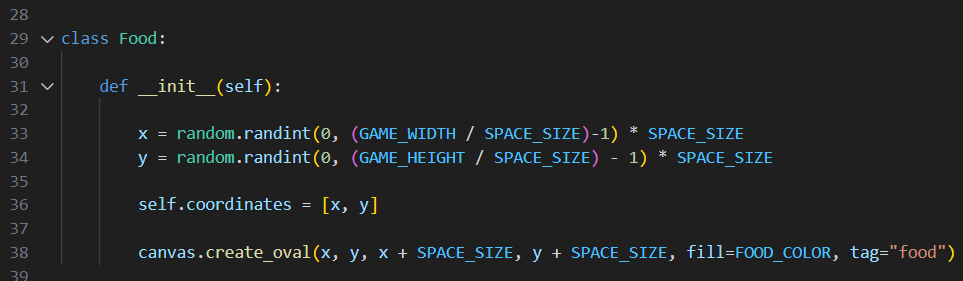


Рисунок 3.

# Пункт 3.

Для определения объекта «змейка» был создан отдельный класс Snake, в котором описано начальная координата движения змейки, её цвет и количество частей (квадратиков), из которых она состоит.

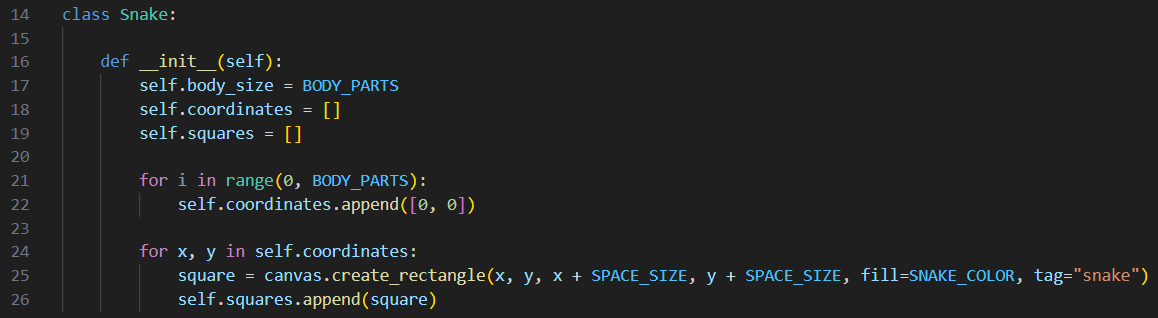


Рисунок 4.

# Пункт 4.

Далее для змейки была создана функция движения, которая работает таким образом, что изначально всё передвижение представляет из себя последовательное увеличение квадратиков, из которых состоит змея (см. рис. 5 ниже).

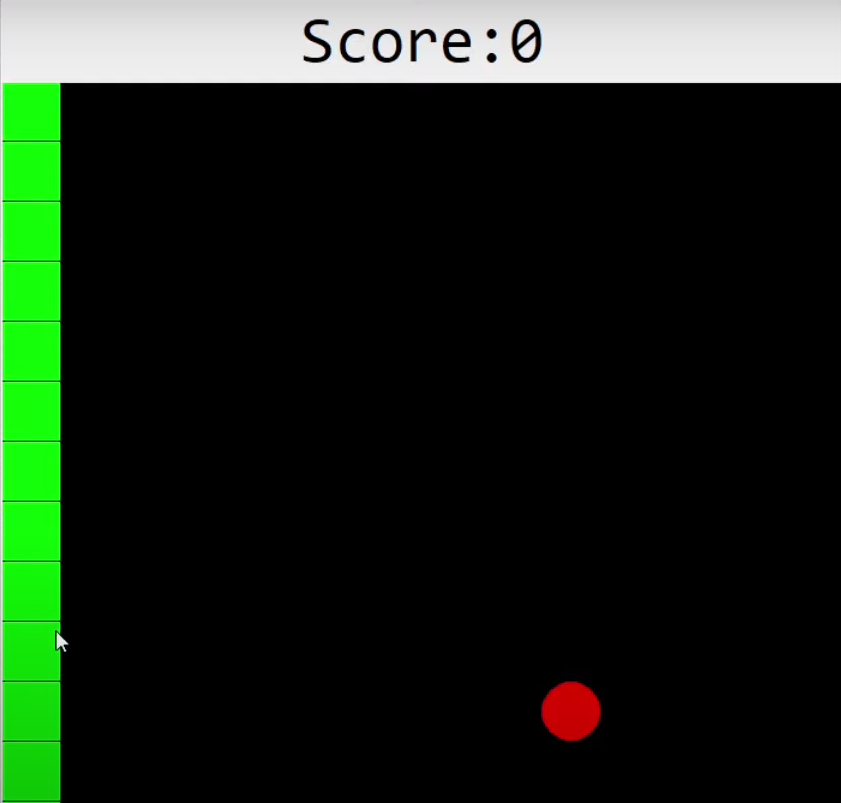


Рисунок 5.

Для реализации игры бесконечная змейка не подходит, поэтому решаю эту проблему удалением лишнего следа из квадратиков. (см. рис. 6 на след. странице).

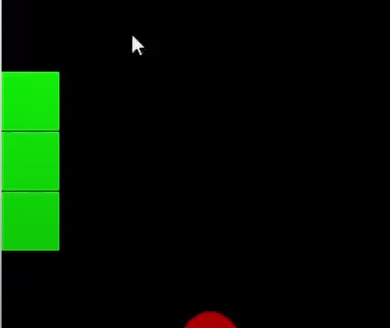


Рисунок 6.

# Пункт 5.

Для задания траектории движения змейки через кнопки на клавиатуре воспользуемся встроенной в библиотеку **tkinter** функцией bind.

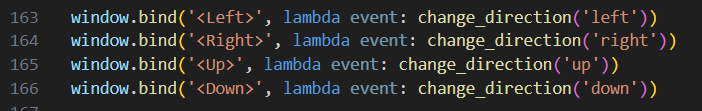


Рисунок 7.

# Пункт 6.

После реализую движение змейки по заданным на клавиатуре кнопкам через старое и новое направление движения и заранее исключаю возможную проблему с движением змейки на 180 градусов в новой функции change\_directions (см. рис. 8 на след. странице).

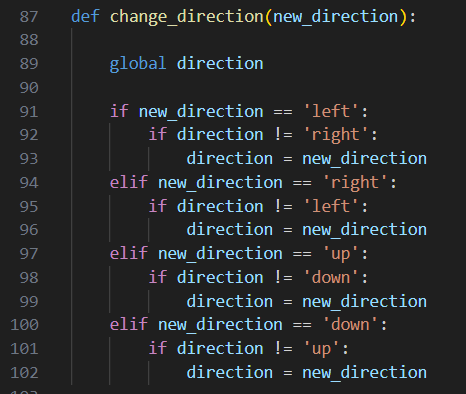


Рисунок 8.

# Пункт 7.

Теперь, когда игра почти готова, можно снова заняться интерфейсом, а конкретнее набором очков в плашке Score и удлинении змеи при съедении яблок.

Условием для набора очков будем считать совпадение координат еды и головы змейки. Т.е., если координаты совпали, то в плашке Score счетчик изменяется на +1. А также не забываем удалять съеденное яблоко и добавлять новое в случайном месте поля игры (см. рис. 9 на след. странице).

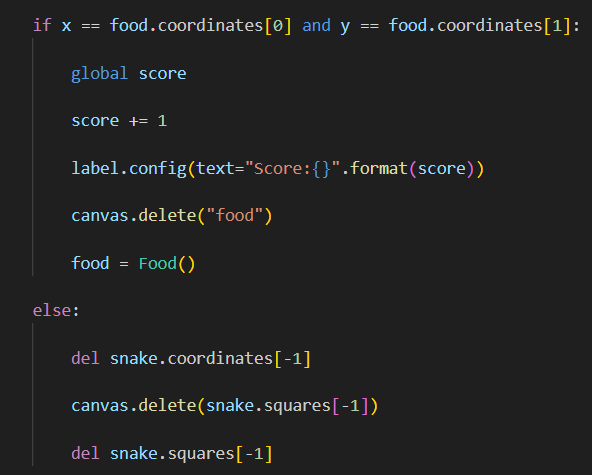


Рисунок 9.

# Пункт 8.

Для ограничения зоны движения змейки была создана отдельная функция с названием check\_collisions, которая проверяет координаты змейки на запрещенные значения, т.е. размеры рамок окна игры. Если эти координаты совпадают, то игра заканчивается. Проверка делается для обоих осей координат.

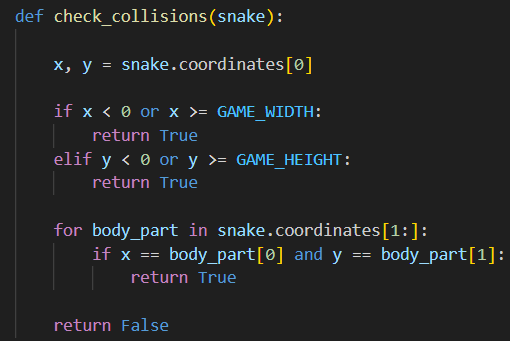


Рисунок 10.

# Пункт 9.

Используя встроенные в библиотеку **tkinter** методы delete, canvas, create\_text была создана функция, при вызове которой на экране появляется сообщение о конце игры.

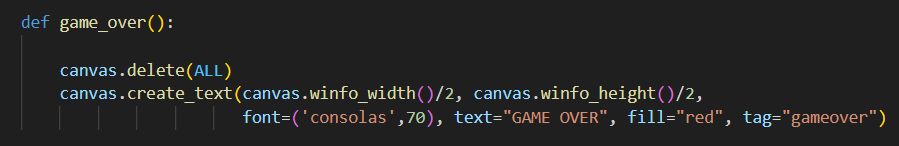


Рисунок 11.

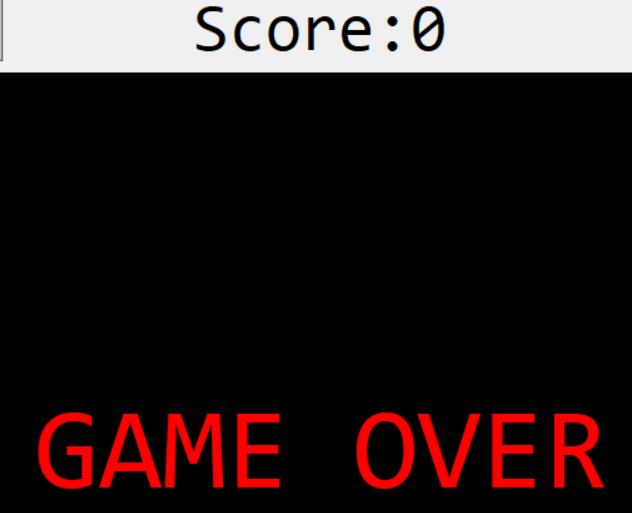


Рисунок 12.

# ЗАКЛЮЧЕНИЕ

В ходе практики была изучена концепция объектно-ориентированного программирования на языке Python, была изучена такая библиотека как **tkinter**, был изучен модуль random и все его функции.

Задачи были направлены на закрепление изученного теоретического материала. Особое внимание уделялось работе со встроенной библиотекой виджетов tkinter и объектно-ориентированному программированию, его принципам и особенностям.

Предлагалось создать классическую игру «Snake» с набором очков, удлинением хвоста змеи и перемещением по ограниченному полу действия игры и её компонентов.

В результате практики были усовершенствованы мои компетенции, закреплены теоретические знания и практические навыки.

В ходе учебной практики я научился: выполнять поставленную задачу, следуя условию; разделять главную задачу на подпункты и определять приоритетность этих пунктов; изменять готовое решение по мере нахождения ошибок, искать альтернативные пути решения задач.

Я изучил библиотеку **tkinter** (расстановка виджетов: изменение их характеристик, приведение их в движение, взаимодействие друг с другом), модуль random (работа со случайными числами, генератор случайностей), принципы ООП.

По мере прохождения учебно-технологической практики я выполнил девять задач, в которых использовал полученные знания и закрепил навыки работы с ЯП Python.

В течение практики все задачи были выполнены, а цели достигнуты.

# СПИСОК ЛИТЕРАТУРЫ

1. Документация синтаксиса ЯП Python - <https://docs.python.org>
2. Документация встроенной библиотеки tkinter - <https://docs.python.org/3/library/tkinter.html>
3. Документация метода random - <https://docs.python.org/3/library/random.html>
4. Объяснение основных методов работы с ООП в Python - <https://cmp.phys.msu.su/sites/default/files/%D0%9E%D0%9E%D0%9F_%D0%BD%D0%B0_Python_%D0%A3%D1%87%D0%B5%D0%B1%D0%BD%D0%BE%D0%B5%20%D0%BF%D0%BE%D1%81%D0%BE%D0%B1%D0%B8%D0%B5_var7.pdf>

# ПРИЛОЖЕНИЕ 1

Код проекта, отчет, дневник - <https://github.com/yellowb3nz/py_game_snake_proj>