

Колганова С.Б.

*студентка факультета естествознания,
математики и информатики
Филиал УРГПУ в г. Нижнем Тагиле
г. Нижний Тагил, Россия*

Научный руководитель:

Бужинская Н.В.

*к.п.н., доцент кафедры информационных технологий
и физико-математического образования
Филиал УРГПУ в г. Нижнем Тагиле
г. Нижний Тагил, Россия*

ПРИМЕНЕНИЕ ЯЗЫКА ПРОГРАММИРОВАНИЯ PYTHON ДЛЯ РАЗРАБОТКИ ИГР

Аннотация

Разработка игр на Python является классическим заданием для изучения основ программирования и алгоритмического мышления. В процессе разработки игры требуется продумать сценарий, спроектировать интерфейс, определить функции игры и обозначить требования к ней. После этих важнейших этапов разработчик приступает к реализации основной идеи игры посредством языка программирования. В статье описан процесс разработки игры «Крестики-нолики» с помощью библиотеки Tkinter.

Ключевые слова: игра, программирование, интерфейс, Python, Tkinter.

Kolganova S.B.

*student of the Faculty of Natural Science,
Mathematics and Computer Science
Branch of RGPPU in Nizhny Tagil,
Nizhny Tagil, Russia*

Scientific supervisor:

Buzhinskaya N.V.

*Candidate of Pedagogical Sciences,
Associate Professor of the Department of Information Technology and Physics
and Mathematics Education
Branch of RGPPU in Nizhny Tagil,
Nizhny Tagil, Russia*

USING THE PYTHON PROGRAMMING LANGUAGE FOR GAME DEVELOPMENT

Annotation

Python game development is a classic assignment for learning the basics of programming and algorithmic thinking. In the process of developing a game, it is necessary to think over the scenario, design the interface, define the functions of the game and identify the requirements for it. After these crucial stages, the developer proceeds to implement the basic idea of the game through a programming language. The article describes the process of developing the game using the Tkinter library.

Keywords: game, programming, interface, Python, Tkinter.

Язык программирования Python — это высокоуровневый, мультипарадигменный и интерпретируемый язык общего назначения, ориентированный на простоту, читаемость кода и высокую производительность разработчика [1]. Его ключевые особенности включают динамическую строгую типизацию, объектно-ориентированность и минималистичный синтаксис, основанный на отступах, что облегчает обучение и делает код лаконичным и понятным. В таблице 1 представлены основные возможности данного языка программирования.

Таблица 1.

Категория возможностей Python

Категория возможностей	Описание и примеры применения
Парадигмы программирования	Императивное, процедурное, структурное, функциональное, объектно-ориентированное, асинхронное, метапрограммирование.
Типизация	Динамическая строгая типизация, что облегчает разработку и повышает гибкость кода.
Автоматическое управление памятью	Сборка мусора освобождает программиста от ручного управления памятью.
Модульность	Поддержка модулей и пакетов для структурирования больших проектов.
Работа с данными и файлами	Обработка XML/HTML, HTTP-запросы, FTP, работа с изображениями, аудио и видео.
Веб-разработка	Создание веб-приложений и API с использованием фреймворков Django, Flask, Pyramid.
Научные вычисления и анализ данных	Библиотеки NumPy, SciPy, Pandas, интеграция с машинным обучением (TensorFlow, Keras, PyTorch).
Обработка естественного	Библиотеки для NLP: NLTK, Gensim и другие.

Категория возможностей	Описание и примеры применения
языка	
Разработка GUI	Создание графических интерфейсов пользователя с помощью Tkinter, PyQt и других библиотек.

Главным преимуществом Python является простота и мощная экосистема с большим количеством готовых библиотек и активным сообществом разработчиков, что облегчает решение сложных задач и обучение основам данного языка. Однако стоит учитывать, что Python уступает в скорости и потреблении памяти языкам низкого уровня и не всегда подходит для разработки требовательных к ресурсам приложений.

Рассмотрим вопросы реализации графического интерфейса в Python с использованием Tkinter

Tkinter — это встроенная стандартная библиотека Python для создания графических пользовательских интерфейсов (GUI), которая позволяет быстро и просто создавать окна и элементы управления, такие как кнопки, текстовые поля, метки и списки [2]. Одним из главных преимуществ Tkinter является его кроссплатформенность — приложения, созданные с её помощью, будут хорошо работать на Windows, Linux и macOS, при этом визуальные элементы адаптируются под стили текущей операционной системы, что придаёт приложению нативный вид.

Для создания GUI с Tkinter, разработчик импортирует её модули, создаёт главное окно (контейнер для всех элементов), после чего добавляет виджеты — базовые строительные блоки интерфейса. Все действия приложения управляются в главном цикле событий (mainloop), который ожидает пользовательский ввод, события мыши и клавиатуры и обновляет интерфейс.

Основные виджеты Tkinter включают кнопки (Button), метки (Label), поля ввода (Entry), списки (Listbox) и выпадающие меню (OptionMenu), что позволяет создавать приложения различной сложности — от простых утилит до полноценных программ с динамическими элементами.

Перейдём к описанию процесса создания игры «Крестики нолики» на языке Python.

Продумывая сценарий, необходимо учитывать, что эта простая логическая игра для двух игроков. Она создаётся как программа в командной строке с полем 3×3. Каждый игрок по очереди ставит свой знак — «X» или «O». При создании игры необходимо подумать, как написать функции для отображения игрового поля, проверки правильности ходов и определения

победителя. Кроме того, необходимо обрабатывать действия игрока, чтобы игра была удобной и понятной.

Таким образом, реализация сценария игры предполагает создание модулей для:

- отображения игрового поля в читаемом виде;
- проверки условий победы по горизонтали, вертикали и диагонали;
- управления логикой игры и сменой ходов.

На первом шаге необходимо создать переменные, позволяющие отслеживать ход игры и обновлять интерфейс при нажатии кнопки (см. листинг 1).

Листинг 1

Создание переменных

```
player_turn = "X"
buttons = []
game_state = [["" for _ in range(3)] for _ in range(3)]
1. Функция создания меню
def create_menu(root):
    menubar = tk.Menu(root)
    root.config(menu=menubar)

    # Меню 1
    file_menu = tk.Menu(menubar, tearoff=0)
    file_menu.add_command(label="Сохранить", command=save_game)
    file_menu.add_command(label="Выйти", command=root.destroy)
    menubar.add_cascade(label="Файл", menu=file_menu)

    # Меню 2
    help_menu = tk.Menu(menubar, tearoff=0)
    help_menu.add_command(label="О программе", command=show_info)
    menubar.add_cascade(label="Справка", menu=help_menu)
```

Далее создадим основу для интерфейса, с которым взаимодействует игрок (см. листинг 2).

Листинг 2

Работа с интерфейсом

```
def create_buttons(root):
    global buttons
    for i in range(3):
        row = []
        for j in range(3):
            button = tk.Button(root, command=lambda row=i,
column=j: click(row, column),
```

```

height=3, width=6,
font=('Helvetica', 24))
    button.grid(row=i, column=j)
    row.append(button)
buttons.append(row)

```

Результат работы представлен на рисунке 1.



Рис 1. Итоговый вид

Далее напишем код для определения победителя (см. листинг 3).

Листинг 3

Определение победителя

```

def click(row, column):
    global player_turn
    if buttons[row][column]['text'] == "":
        buttons[row][column]['text'] = player_turn
        game_state[row][column] = player_turn
        if check_win():
            messagebox.showinfo("Игра окончена", f"Победил игрок {player_turn}!")
            root.destroy()
            player_turn = "0" if player_turn == "X" else "X"
def check_win():
    # Проверка строк и столбцов
    for i in range(3):
        if game_state[i][0] == game_state[i][1] == game_state[i][2] != "":
            return True
        if game_state[0][i] == game_state[1][i] == game_state[2][i] != "":

```

```
return True
```

```
# Проверка диагоналей
if game_state[0][0] == game_state[1][1] == game_state[2][2] !=
"":
    return True
if game_state[0][2] == game_state[1][1] == game_state[2][0] !=
"":
    return True
return False
```

Результат работы представлен на рисунке 2.



X	O	X
X	O	O
	O	X

Рис 2. Определение победителя

Далее добавим функцию для отображения информации о победителе (см. листинг 4, рис. 3).

Листинг 4

Отображение информации о программе

```
def save_game():
    with open("save.txt", "w") as f:
        for row in game_state:
            f.write(" ".join(row) + "\n")
    messagebox.showinfo("Сохранение", "Игра сохранена в файл save.txt")
def show_info():
    info = "Игра Крестики-нолики\nЖиткова Софья\nПроект по программированию"
    messagebox.showinfo("О программе", info)
root = tk.Tk()
root.title("Крестики-нолики")
root.geometry("300x350")
create_menu(root)
create_buttons(root)
```

```
root.mainloop()
```

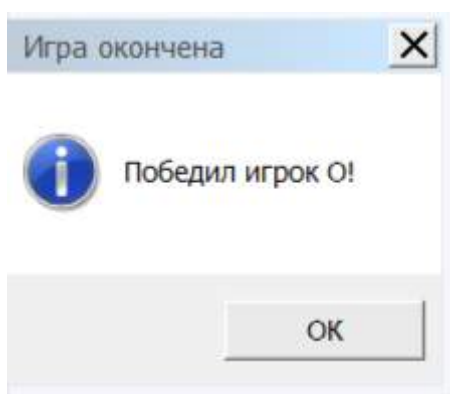


Рис 3. Отображение информации о победителе

В этой статье была создана игра «Крестики-нолики» на языке Python. Программа позволяет двум игрокам делать ходы, определяет победителя и следит за правильностью ходов. Игра помогает понять основы программирования на Python и может быть хорошим первым проектом для новичков. В будущем её можно улучшить, добавив в качестве игрока компьютер.

СПИСОК ИСТОЧНИКОВ

1. Воронов, М. В. Системы искусственного интеллекта : учебник и практикум для среднего профессионального образования / М. В. Воронов, В. И. Пименов, И. А. Небаев. — Москва : Издательство Юрайт, 2025. — 268 с.
2. Гниденко, И. Г. Технологии и методы программирования : учебник для вузов / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — Москва : Издательство Юрайт, 2025. — 241 с.
3. Гниденко, И. Г. Технология разработки программного обеспечения : учебник для среднего профессионального образования / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — Москва : Издательство Юрайт, 2025. — 248 с.
4. Федоров, Д. Ю. Программирование на python : учебное пособие для вузов / Д. Ю. Федоров. — 6-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 187 с.
5. Чернышев, С. А. Основы программирования на Python : учебник для вузов / С. А. Чернышев. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 349 с.