

Л. М. Коренюгина

Сибирский федеральный университет, г. Красноярск, Россия

Средняя школа № 63, г. Красноярск, Россия

РАЗРАБОТКА УЧЕБНЫХ ПРОЕКТОВ В 10–11 КЛАССАХ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ TKINTER: МЕТОДИКА И ПРАКТИКА

Аннотация

В статье рассматриваются методические и практические аспекты использования библиотеки Tkinter языка программирования Python при организации проектной деятельности в 10–11 классах. В статье представлена методика разработки учебных проектов на основе библиотеки Tkinter, которая рассматривается как эффективное средство интеграции знаний и формирования универсальных учебных действий. Показано, как создание графических приложений на основе Tkinter способствует развитию алгоритмического и вычислительного мышления, повышает мотивацию учащихся и обеспечивает реализацию деятельностного подхода в обучении информатике. Проектирование интерфейсов способствует развитию у школьников навыков координатного, алгоритмического и пользовательского мышления. Апробация методики проводилась в 1 полугодии 2025-2026 учебного года на базе средней школы 63 г. Красноярска с участием учащихся 10–11 классов. По результатам анкетирования учащихся уровень мотивации к изучению информатики увеличился на 34%. В статье представлены примеры учебных проектов, описаны этапы их разработки, предложены рекомендации по дифференциации заданий и интеграции с ключевыми темами курса. Показано, что метод `place()` в Tkinter представляет собой не просто технический инструмент позиционирования, а мощный педагогический ресурс для формирования у учащихся координатного мышления. Материал ориентирован на учителей, стремящихся эффективно использовать проектную деятельность для формирования цифровой грамотности и для подготовки обучающихся к выполнению индивидуального итогового проекта.

Ключевые слова: Tkinter, Python, проектная деятельность, графический интерфейс, учебный проект, индивидуальный итоговый проект.

Контактная информация

Коренюгина Людмила Михайловна,

старший преподаватель базовой кафедры интеллектуальных систем управления, Институт космических и информационных технологий, Федеральное государственное автономное образовательное учреждение высшего образования «Сибирский федеральный университет», г. Красноярск, Россия; *адрес:* 660074, г. Красноярск, ул. Киренского, 26Б, корпус № 17 (Ж); Россия;

учитель информатики, Муниципальное автономное общеобразовательное учреждение «Средняя школа №63», г. Красноярск, Россия; *адрес:* 660059, г. Красноярск, ул. Вавилова, д. 49Б;
e-mail: korenugina71@yandex.ru

L. M. Korenyugina,

Siberian Federal University, Krasnoyarsk, Russia

Secondary school No. 63, Krasnoyarsk, Russia

DEVELOPING CURRICULUM PROJECTS IN GRADES 10–11 USING THE TKINTER LIBRARY: METHODOLOGY AND PRACTICE

Abstract

This article examines the methodological and practical aspects of using the Tkinter library of the Python programming language to organize project-based activities in grades 10 and 11. It demonstrates how creating graphical applications based on Tkinter promotes algorithmic and computational thinking, increases student motivation, and supports an activity-based approach to teaching computer science. Examples of educational projects are presented, the stages of their development are described, and recommendations for differentiating assignments and integrating them with key course topics are offered. It is shown that the `place()` method in Tkinter is not just a technical positioning tool, but a powerful pedagogical resource for developing students' coordinate thinking. The material is intended for teachers seeking to effectively employ project-based learning to develop students' digital literacy and prepare them for completing an individual capstone project.

***Keywords:* Tkinter, Python, project work, graphical interface, educational project, individual final project.**

1. Введение

Современный курс информатики в старшей школе делает акцент на системно – деятельностном подходе, проектной деятельности и формировании цифровой грамотности [1, 2]. Одной из ключевых задач учителя становится не просто передача знаний, а создание условий для самостоятельного проектирования, реализации и презентации программных продуктов. Создание собственных программных продуктов с графическим интерфейсом повышает мотивацию школьников и делает обучение программированию осмысленным и практико-ориентированным.

Библиотека Tkinter представляет собой доступный и эффективный инструмент для организации такой деятельности в школьной среде [3]. Библиотека Tkinter входит в стандартную поставку языка программирования Python и не требует установки дополнительных модулей, что особенно важно в условиях ограниченных технических ресурсов многих образовательных

организаций. Библиотека обладает простым синтаксисом, поддерживает основные виджеты (кнопки, поля ввода, метки, холст) и позволяет за несколько уроков перейти от теоретических основ к созданию функциональных приложений [4].

Tkinter позволяет реализовать индивидуальные итоговые проекты (ИИП), предусмотренные ФГОС СОО: от простого калькулятора до персонального органайзера или учебного тренажёра. При этом проекты остаются технически выполнимыми даже для учащихся с базовым уровнем подготовки.

2. Формирование метапредметных умений при работе с Tkinter

Одной из ключевых целей современного образования в соответствии с ФГОС является развитие метапредметных результатов — универсальных способов деятельности, применимых в различных учебных и жизненных ситуациях [5]. Проектная работа с использованием библиотеки Tkinter выступает эффективной средой для формирования таких умений, выходящих за рамки чисто технического программирования.

Разработка приложений с использованием библиотеки Tkinter становится эффективной средой для формирования вычислительного мышления — совокупности когнитивных навыков, позволяющих решать задачи с помощью компьютерных методов [6].

На этапе проектирования простого GUI-приложения учащийся вынужден:

- 1) декомпозировать задачу на логические блоки (ввод → обработка → вывод);
- 2) формализовать поведение программы (например: «при нажатии кнопки вызывается функция расчёта»);
- 3) анализировать возможные сценарии, включая ошибочные (пустой ввод, некорректные данные), и предусматривать их обработку.

Такой подход развивает у школьников алгоритмическую культуру, логическую дисциплину и умение мыслить системно [7]. Данные качества, востребованы не только в программировании, но и в научной, инженерной, экономической и повседневной деятельности [8].

Проектное мышление.

Работа над GUI-приложением представляет собой полноценный учебный мини-проект, включающий следующие этапы:

- 1) Постановка цели и определение требований. Учащийся формулирует, какую задачу решает приложение и какие функции оно должно выполнять;
- 2) Проектирование интерфейса. Обычно на начальном этапе выполняется в виде эскиза на бумаге или цифрового макета;
- 3) Итеративная реализация и тестирование. Данный этап включает написание кода с постепенным добавлением функциональности и проверку корректности работы;
- 4) Презентацию и защиту результата. На этом этапе проводится демонстрация готового

продукта перед аудиторией с объяснением архитектуры и принятых решений.

Реализация всех этапов формирует у учащихся понимание жизненного цикла программного продукта и навыки управления собственной учебной деятельностью, способствует формированию как предметных, так и метапредметных компетенций [9].

Пользовательская ориентация и эмпатия.

В процессе проектирования учащийся начинает должен учитывать аспекты, как удобство навигации, читаемость надписей, логичность расположения элементов управления и визуальная ясность интерфейса. При работе с инструментом Tkinter школьники осваивают основы пользователь–ориентированного дизайна. Этот подход, при котором главным критерием качества интерфейса становится удобство и понятность для конечного пользователя [10].

Критическое мышление и саморегуляция.

Процесс разработки GUI–приложения на основе Tkinter неизбежно включает отладку кода, поиск причин сбоев и сравнение альтернативных решений (например, выбор между методами размещения виджетов — .pack(), .grid() или .place()) и принятие обоснованных решений. Цикл «проблема – гипотеза – проверка – вывод» способствует формированию критического мышления, рефлексивности и саморегуляции учебной деятельности [11]. Школьник постепенно переходит от пассивного исполнения инструкций к осознанному, самостоятельному обучению — учится не просто «делать, как сказано», а думать, пробовать, анализировать и учиться на собственном опыте [12].

Развитие коммуникативных и презентационных навыков.

Создание графического приложения с помощью библиотеки Tkinter — это не только техническая задача, но и важный этап формирования коммуникативной компетентности учащихся [13]. Поскольку результат проекта имеет визуальную и функциональную форму, его необходимо не просто написать, но и представить, объяснить и защитить. Защита учебного проекта — это целостный образовательный этап, в ходе которого учащийся демонстрирует не только техническую компетентность, но и уровень сформированности коммуникативных, презентационных и рефлексивных умений [14].

Типичная защита проекта включает в себя следующие этапы:

1) Введение и постановка задачи.

Учащийся формулирует цель проекта, обосновывает его актуальность и описывает целевую аудиторию. На данном этапе формируется умение чётко определять проблему и аргументировать выбор темы.

2) Презентация архитектуры и функционала.

На данном этапе учащийся раскрывает архитектуру своего приложения: поясняет, какие виджеты Tkinter используются и как они организованы, описывает логику обработки

пользовательских данных и объясняет, как осуществляется взаимодействие между элементами графического интерфейса. На данном этапе у учащихся формируется умение переводить технические решения на доступный язык, выделять ключевые аспекты и логически структурировать информацию [15].

3) Демонстрация работы приложения.

Учащийся осуществляет демонстрацию работы приложения: вводит исходные данные, взаимодействует с элементами графического интерфейса (например, активирует кнопки) и наглядно демонстрирует, как программа обрабатывает информацию и выводит результат. Этот этап способствует формированию навыков публичного выступления, умения управлять вниманием аудитории, сохранять коммуникативную уверенность и профессионально представлять собственный продукт — качества, востребованные как в учебной, так и в будущей профессиональной деятельности [16].

4) Ответы на вопросы и рефлексия.

Учащийся отвечает на вопросы и замечания аудитории, критически анализирует сильные и слабые стороны своего проекта, а также предлагает возможные пути его доработки и улучшения.

Этот этап способствует развитию критического мышления, эмоциональной устойчивости и готовности воспринимать обратную связь как ценный ресурс для личного и профессионального роста.

3. Типология проектов

Для реализации дифференцированного подхода в обучении программированию с использованием библиотеки Tkinter предложена трёхуровневая типология учебных проектов, отражающая поступательное усложнение как технических, так и когнитивных задач [17].

Базовый уровень ориентирован на освоение основ графического интерфейса: учащиеся создают простые приложения с вводом, обработкой и выводом данных (например, калькулятор суммы, бинарный калькулятор), используя виджеты Entry, Button, Label и базовую обработку исключений.

Средний уровень предполагает работу с логикой принятия решений и управлением состоянием программы: типовой проект — электронный билет на самолет, включающий Radiobutton, списки, базу данных и условные конструкции.

Продвинутый уровень включает взаимодействие с внешними ресурсами и более сложную архитектуру: например, интернет магазин книг и оформление кассового чека о покупке с использованием Listbox, Text, функций и работы с файлами (чтение/запись).

Такая типология позволяет каждому учащемуся работать в своей зоне ближайшего развития, последовательно переходя от линейных программ к модульным решениям, и создаёт прочную основу для выполнения индивидуального итогового проекта.

4. Практические примеры

Ниже представлены примеры учебных проектов, реализуемых на разных уровнях сложности. Каждый из них доступен для самостоятельной разработки в течение нескольких уроков и может быть адаптирован под интересы учащихся (например, конвертер валют вместо бинарного калькулятора).

Пример 1 Базовый уровень: Калькулятор суммы двух чисел

Проект представляет собой простое графическое приложение, позволяющее ввести два числа и получить их сумму. Проект охватывает ключевые аспекты GUI-программирования на начальном этапе. Такой проект закрепляет навыки работы с виджетами, обработкой событий и исключениями. Основными элементами интерфейса являются два поля ввода (Entry) для чисел, кнопка «Сложить» и метка (Label) для вывода результата или сообщения об ошибке. На рисунке 1 изображен проект интерфейса калькулятора суммы двух чисел на Python.

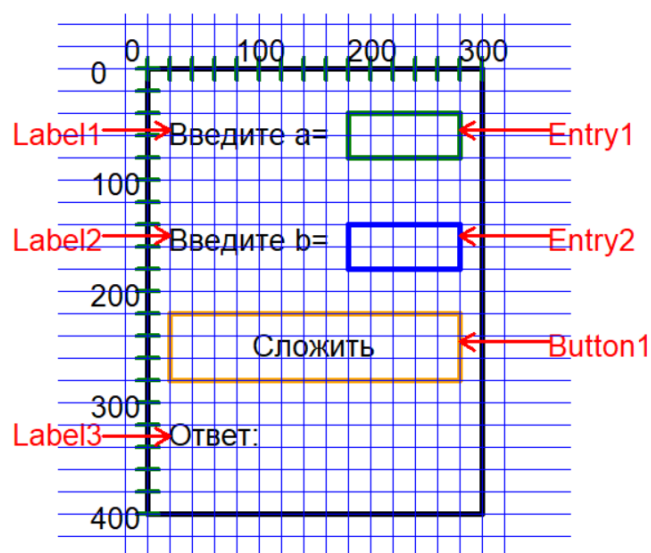


Рис. 1. Проект интерфейса калькулятора суммы двух чисел

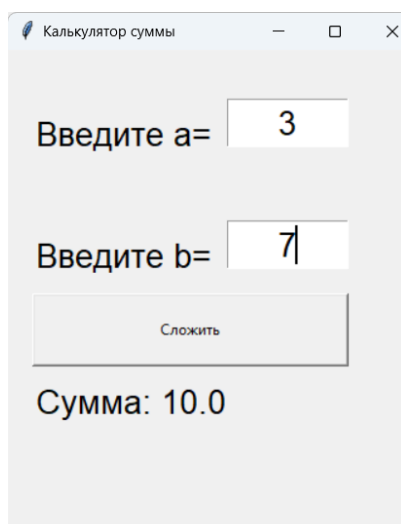


Рис. 2 Результат работы программы «Калькулятор суммы двух чисел»

Код программы

```

#Импортировать библиотеку Tkinter:
importtkinter as tk
def calculate():
try:
    a = float(entry1.get())
    b = float(entry2.get())
label3.config(text=f"Сумма: {a+b}")
exceptValueError:
label3.config(text="Ошибка: введитечисла")
#Инициализировать главное окно:
root = tk.Tk()
root.title("Калькулятор суммы")
root.geometry("340x400")
#Шаг 2. Размещение виджетов
#Создать метки и поля ввода:
Label1=tk.Label(root, text="Введитеa=", font=('Arial',20))
Label1.place(x=20,y=50)

entry1 = tk.Entry(root, font=('Arial',20),justify='center')
entry1.place(x=180,y=40,width=100,height=40)
label1=tk.Label(root, text="Введите b=", font=('Arial',20))
label1.place(x=20,y=150)
entry2 = tk.Entry(root, font=('Arial',20),justify='center')
entry2.place(x=180,y=140,width=100,height=40)

#Добавить кнопку и метку для результата:
button1 = tk.Button(root, text="Сложить", command=calculate)
button1.place(x=20,y=200,width=260,height=60)
label3=tk.Label(root, text="Ответ:", font=('Arial',20))
label3.place(x=20,y=270)
root.mainloop()

```

С помощью дополнительных заданий на изменение цвета кнопок, фона, типа шрифтов можно развить навыки настройки и адаптации графического интерфейса и понимание принципов визуального дизайна.

Педагогическая ценность проекта заключается в том, что он позволяет за один урок перейти от теории к наглядному результату, формируя мотивацию и уверенность в своих силах.

Пример 2. Бинарный калькулятор

Задание: Разработать графическое приложение на языке Python с использованием библиотеки Tkinter, которое преобразует целое неотрицательное число из десятичной системы счисления в двоичную.

```

23: 2 = 11 остаток 1
11: 2 = 5 остаток 1
5:2 = 2 остаток 1
2 : 2 = 1 остаток 0
1: 2 = 0 остаток 1

```

Результат: $13_{10} = 10111_2$

Команда `bin(23)` возвращает `'0b10111'`.

Префикс `0b` указывает на двоичную систему. Чтобы получить «чистое» двоичное число, используем срез строки: `bin(23)[2:]` → `'10111'`.

Интерфейс калькулятора представлен на рисунке 3 и содержит следующие элементы (виджеты):

- 1) Окно размером 500×300 пикселей
- 2) Метка «Десятичное число a=» (шрифт Arial, размер 20) в левой части окна
- 3) Поле ввода для десятичного числа (ширина 100px, высота 40px) справа от метки
- 4) Кнопка «Перевести» (шрифт Arial, размер 24) под полями ввода, занимающая почти всю ширину окна
- 5) Метка для отображения результата «Двоичное число b=» под кнопкой.



Рис.3. Проект интерфейса «Бинарный калькулятор»

Результат работы программы представлен на рисунке 4.

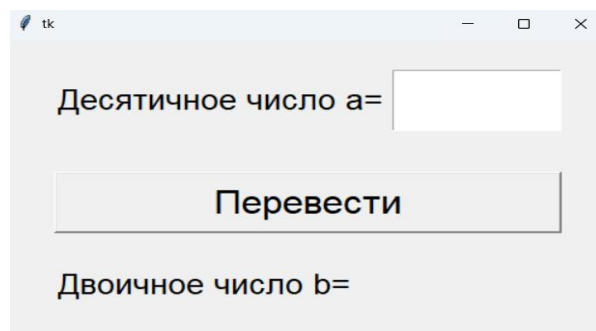


Рис. 4. Результат работы программы «Бинарный калькулятор»/ Fig. 4. The result of the “Binary calculator” program

Опишем функционал программы.

При нажатии кнопки «Перевести»:

- 1) Считать значение из поля ввода;
- 2) Преобразовать строку в целое число;
- 3) Перевести число в двоичную систему с помощью `bin()`;
- 4) Убрать префикс `0b` с помощью среза `[2:]`;

5) Вывести результат в формате: Двоичное число b=101010.

Программа должна корректно обрабатывать числа от 0 до 1 000 000. Ниже приведен код программы.

Код программы:

```
import tkinter as tk
from tkinter import ttk
root = tk.Tk()
root.geometry('500x300')
def funk():
    a = int(entry1.get())
    b = str(bin(a)[2:])
    label2.config(text=f'Двоичное число b={b}')
    label1 = tk.Label(root, text="Десятичное число a=", font=("Arial", 20))
    label1.place(x=40, y=40)
    entry1 = tk.Entry(root, font=("Arial", 20))
    entry1.place(x=320, y=30, width=140, height=60)
    button1 = tk.Button(root, text="Перевести", font=("Arial", 24), command=funk)
    button1.place(x=40, y=140, width=420, height=60)
    label2 = tk.Label(root, text="Двоичное число b=", font=("Arial", 20))
    label2.place(x=40, y=220, height=40)
root.mainloop()
```

Для повышения сложности задания предлагаем спроектировать калькулятор систем счисления с произвольным основанием. Он осуществляет перевод чисел между десятичной, двоичной, восьмеричной и шестнадцатеричной системами и использует виджеты: Entry, Button, Label, Frame. Нужно предусмотреть обработку ошибок (некорректный ввод). Задание развивает навыки проектирования интерфейсов с учётом пользовательского опыта (автоматическое обновление списков, валидация) и позволяет продемонстрировать работу со словарями и условной логикой. Дополнительные задания позволяют дифференцировать обучение — базовый вариант выполнит любой ученик, расширенный — проявит творческий подход сильных учащихся.

Пример 4. Приложение «Билет на самолет».

Учащийся создаёт приложение для расчета стоимости билета на самолет, используя виджеты Combobox, Radiobutton и переменные StringVar. После завершения выводится расстояние между городами и стоимость билета с учетом выбора класса поездки. При нажатии кнопки «Купить билет» данные поездки передаются в текстовый файл, формируя электронный билет. Проект развивает умение работать со списками, логическими условиями и пользовательским вводом, а также с файлами. Используются виджеты Combobox, Button, Entry, а также функции open(), read(),

write(). Такой проект интегрирует темы «Файловая система» и «Структуры данных». Проект интерфейса приложения представлен на рисунке 5. Результат работы программы представлен на рисунке 6.

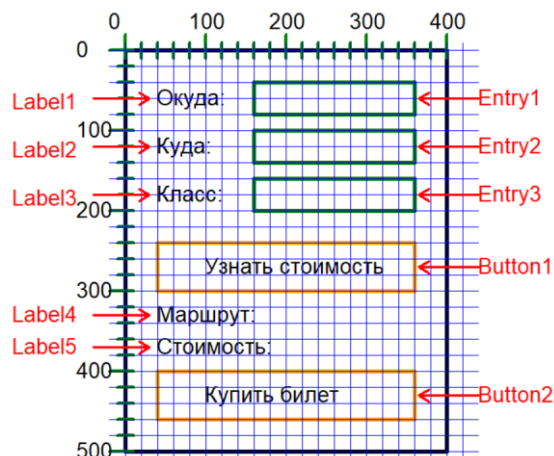


Рис. 5. Проект интерфейса приложения "Билет на самолет"

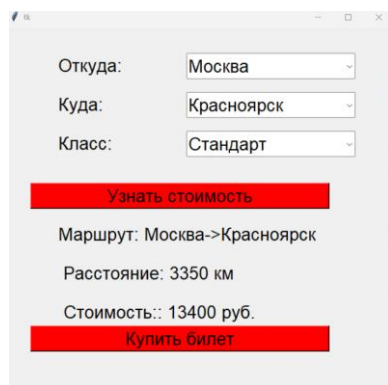


Рис. 6. Результат работы программы «Билет на самолет»

Задачи для самостоятельного выполнения проектов на Tkinter (10–11 класс)

1. *Калькулятор площади по формуле Герона.* Программа принимает три числовых значения (стороны треугольника), проверяет существование треугольника, вычисляет его площадь по формуле Герона и периметр. Результат выводится в удобном формате.

2. *Конвертер миль/км.* Программа для перевода расстояния между милями и километрами с двусторонней конвертацией, возможностью выбора направления и точности вычислений.

3. *Геометрический калькулятор координат.* Программа позволяет вычислять площади геометрических фигур, заданных координатами вершин на плоскости. Пользователь вводит координаты точек, программа визуализирует фигуру на координатной сетке и вычисляет её площадь по математическим формулам.

4. Развитие координатного мышления через выбор методов позиционирования объектов.

Координатное мышление — это способность воспринимать пространство как систему отсчёта, где каждый объект имеет точное положение (координаты), а перемещение/изменение описывается математическими законами. Принципиальное отличие экранной системы координат

от декартовой плоскости влияет на размещение виджетов на форме и может использоваться как педагогический инструмент. Несмотря на внешнюю аналогию (обе системы используют пару чисел (x, y) для определения положения точки), экранная система координат содержит критическое отличие, которое формирует базовое мышление программиста и влияет на все аспекты визуальной разработки. При размещении виджетов с помощью метода `place(x, y)` учащиеся оперируют привычными математическими понятиями: абсцисса — координата x (горизонтальное смещение от левой границы окна); ордината — координата y (вертикальное смещение от верхней границы окна). Прямая аналогия с декартовой системой координат позволяет применить знания из курса алгебры и геометрии на практике: при проектировании интерфейса учащиеся рассчитывают координаты элементов, используют формулы арифметической прогрессии для равномерного вертикального расположения ($y = y_0 + i \cdot d$), а также работают с понятием точки привязки (`anchor`). Метод `place()` по сравнению с методами `grid()` и `pack()` в учебном контексте обладает уникальным педагогическим потенциалом: делает видимой и осязаемой ту координатную логику, которую автоматизированные менеджеры компоновки скрывают. Отказ от `place()` на первоначальном этапе обучения означает утрату возможности сформировать у учащихся целостное представление о пространственных системах координат — фундаментальном понятии как для информатики, так и для смежных дисциплин. Менеджеры компоновки `grid()` и `pack()` скрывают координатную логику, автоматизируя размещение виджетов. Они эффективны в промышленной разработке для адаптивных интерфейсов. Однако в учебном контексте их автоматизация лишает учащегося возможности осознать, как именно размещаются элементы на экране. Для учебных целей при изучении систем координат это означает потерю педагогического эффекта — учащийся не видит и не осознаёт работу с физическими координатами экрана.

5. Выводы

Библиотека Tkinter, входящая в стандартную поставку Python, представляет собой эффективный инструмент для реализации проектной деятельности в старших классах. Использование библиотеки Tkinter в 10–11 классах — это педагогическая стратегия, направленная на развитие вычислительного мышления, творческой самостоятельности и готовности к реальной разработке программного обеспечения. Применение метода `place()` в библиотеке Tkinter представляет собой педагогически оправданный выбор, поскольку обеспечивает глубокую содержательную связь с курсом математики и формирует фундаментальные представления о пространственных системах координат. Он превращает абстрактное понятие «координатная плоскость» в инструмент проектирования, формирует пространственное мышление и создаёт прочную основу для последующего изучения графики, анимации и игровой разработки. Предложенная методика может быть рекомендована к широкому применению в образовательной

практике.

Список источников

1. Акмолдина А. И., Абикинова Т. З., Сейдахметов Б. Возможности модуля Tkinter. // Глобальные проблемы модернизации национальной экономики: материалы XI Международной научно-практической конференции. Тамбов, 2022. С. 145–150. EDN: EJVSFC.
2. Барганалиева Ж. К., Садырова М. Р., Асанбекова Н. О. Создание интерактивного теста на Tkinter в Python. // Вестник Кыргызского государственного университета имени И. Арабаева. 2024. № 1. С. 286–294. EDN: AJUQFV. DOI: 10.33514/1694–7851–2024–1–286–294. EDN: AJUQFV.
3. Босова Л. Л. О новых подходах к изучению школьной информатики в условиях цифровой трансформации общества. // Информатика в школе. 2022. № 4. С. 5–14. EDN: DKRLZV. DOI: 10.32517/2221–1993–2022–21–4–5–14
4. Быкова К. И., Кузьмичева Е. А., Михайлова М. Г. Влияние изучения языка программирования Python на развитие алгоритмического мышления. // Экономика. Общество. Человек : материалы национальной научно-практической конференции с международным участием (Белгород, 18–19 мая 2023 г.) / Белгородский государственный технологический университет им. В. Г. Шухова. Белгород: БГТУ им. В. Г. Шухова, 2023. С. 27–34. МАОУ «Воронежский государственный педагогический университет». EDN: DEKQNB.
5. Гаркавенко Г. В., Бакулина Ю. С., Сильвестров И. Е. Моделирование текстового интерфейса средствами языка Python. // Информатика в школе. 2023. № 5. С. 57–61. DOI: 10.32517/2221–1993–2023–22–5–57–61
6. Добровольская Н. Ю., Харченко А. В. Задания–исследования как способ развития навыков программирования. // Информатика в школе. 2019. № 3. С. 48–51. DOI: 10.32517/2221–1993–2019–18–3–48–51
7. Ефремов Е. А. Разработка графических интерфейсов с помощью графической библиотеки Tkinter. // Молодёжный научно-технический вестник. Академия инженерных наук им. А. М. Прохорова. 2017. № 6. С. 44. EDN: ZTUWCB.
8. Жорняк А. Г., Морозова Т. А. Разработка визуального пользовательского интерфейса в программах для научных и инженерных вычислений на языке программирования Python. Часть IV. Библиотека Tkinter. // Научно-технический вестник Поволжья. 2024. № 1. С. 73–76. EDN: QZULMG.
9. Каракозов С. Д., Маняхина В. Г. Python как базовый язык обучения программированию в школе. // Информатика в школе. 2020. № 1. С. 26–30. DOI: 10.32517/2221–1993–2020–19–1–26–30
10. Кривоплясова Е. В., Нефёдова В. Ю., Прилепина А. В. Методика обучения основам программирования на языке Python. // Информатика в школе. 2020. № 3. С. 24–30. DOI: 10.32517/2221–1993–2020–19–3–24–30

11. *Лаухин В. В.* О создании программ на языке Python с помощью графической библиотеки Tkinter. // Системы управления, технические системы: устойчивость, стабилизация, пути и методы исследования: материалы научно-практического семинара молодых учёных и студентов. Елец, 2017. С. 123–128. EDN: XUXITB.
12. *Левашов М. А.* Создание приложения «Усложнённые крестики-нолики» на языке Python. // Научные высказывания. 2024. № 13. С. 15–20. EDN: ARDQQB.
13. *Лужков А. А., Тюканов А. С.* Основы работы в Python : учебный компьютерный практикум Российский государственный педагогический университет им. А. И. Герцена, 2022. 76 с. EDN: GGUDAZ.
14. *Новичихина А. А.* Разработка простых приложений средствами Tkinter в Python. // Актуальные исследования. 2023. №4. С. 32–35. EDN: YJQOAR.
15. *Радионов С. В.* Создание интерфейса для программы Python 3 с помощью библиотеки Tkinter. // Постулат. Приамурский государственный университет им. Шолом-Алейхема. 2019. № 1. С. 39–41. EDN: VWZZAG.
16. *Романов Д. А.* Создание графического интерфейса программы с помощью Tkinter. // Постулат. Приамурский государственный университет им. Шолом–Алейхема. Биробиджан. 2022. № 6. С. 80–100. EDN: IEZMTA.
17. *Фармонов Т. Т.* Графический интерфейс пользователя в Python. // Фундаментальные и прикладные научные исследования в современном мире: сборник научных статей по материалам IV Международной научно-практической конференции (Уфа, 22 марта 2024 г.). 2024. С. 114–119. EDN: FGXAAJ.