УДК 373.5:004.43

Лысова Юлия Леонидовна

учитель информатики

Частное образовательное учреждение «Газпром школа Санкт-Петербург»

ПРАКТИКО-ОРИЕНТИРОВАННАЯ МЕТОДИКА ОБУЧЕНИЯ ПРОГРАММИРОВАНИЮ В СТАРШЕЙ ШКОЛЕ: РАЗРАБОТКА ПОДПРОГРАММ В ФОРМАТЕ КОМАНДНОГО ПРОЕКТА

Аннотация. В работе обобщен опыт проведения практического занятия по информатике в 10-м классе, в ходе которого школьники осваивали написание подпрограмм на языке Python через участие в коллективной разработке многофункционального калькулятора. Каждому участнику предлагалось создать отдельную функцию, проверить её работоспособность и передать код для объединения в общий проект. Концептуальная особенность методики заключается в соединении персональной ответственности за собственный модуль с совместной деятельностью по достижению итогового результата. В публикации поэтапно описывается организация занятия, приводятся используемые цифровые средства сбора материалов, раскрываются уровни сложности предлагаемых заданий и критерии их оценки. Показано, что вовлечение школьников в создание действующего приложения наполняет изучение подпрограмм практическим содержанием, способствует выработке навыков тестирования и отладки, а также создаёт условия для становления регулятивных и коммуникативных универсальных учебных действий. Изложенная модель может быть адаптирована к другим разделам школьного курса программирования.

Ключевые слова: информатика, обучение программированию, Python, подпрограммы, функции, практико-ориентированное обучение, проектная деятельность, коллективная работа, программный продукт, цифровые образовательные средства.

Abstract. The paper summarizes the experience of conducting a practical computer science lesson in the 10th grade, during which students mastered writing subprograms in Python through participation in the collective development of a multifunctional calculator. Each participant was invited to create a separate function, check its operability, and transfer the code for integration into a common project. The conceptual feature of the methodology lies in the combination of personal responsibility for one's own module with joint activities to achieve the final result. The publication describes step by step the organization of the lesson, provides the digital tools used for collecting materials, reveals the levels of task complexity and their assessment criteria. It is shown that involving students in the creation of a working application fills the study of subprograms with practical content, contributes to the development of testing and debugging skills, and creates conditions for the development of regulatory and communicative universal learning actions. The described model can be adapted to other sections of the school programming course.

Keywords: computer science, programming education, Python, subprograms, functions, practice-oriented learning, project-based activity, teamwork, software product, digital educational tools.

Введение

В последние годы школьное образование претерпевает заметные изменения. Всё большее значение придаётся не столько объёму усвоенной информации, сколько умению применять полученные знания в нестандартных ситуациях, выстраивать индивидуальную стратегию действий и работать в сотрудничестве с другими людьми. Эти установки находят отражение в требованиях федерального государственного образовательного стандарта среднего общего образования, который среди приоритетных называет способность обучающихся самостоятельно определять цели, планировать этапы их достижения, контролировать и корректировать собственную деятельность, а также участвовать в коллективных проектах [1].

Научная новизна исследования состоит в разработке и апробации методики краткосрочной командной разработки программного продукта в рамках одного урока, что позволяет преодолеть разрыв между формальным изучением синтаксиса и формированием навыков проектной деятельности в условиях ограниченного времени.

Выбор методов педагогического наблюдения и анализа продуктов деятельности обусловлен необходимостью фиксации как процессуальных (ход работы, взаимодействие), так и результативных (качество кода, уровень самостоятельности) аспектов учебной деятельности в условиях ограниченного времени урока.

Указанные ориентиры в полной мере относятся к преподаванию программирования - области, где формальное владение синтаксисом не равнозначно пониманию процессов конструирования работающих приложений. В нашей практике мы неоднократно сталкивались с ситуацией: ученики успешно справляются с отдельными задачами, демонстрируют знание команд и конструкций языка, но испытывают заметные трудности, когда от них требуется интегрировать написанный фрагмент в структуру функционирующей программы. Особенно отчётливо этот разрыв проявляется при прохождении темы, посвящённой подпрограммам. Освоение механизмов описания функций, передачи аргументов и возврата значений нередко замыкается на решении серии несвязанных упражнений, что не даёт учащимся целостного представления о роли этих конструкций в архитектуре приложения.

Таким образом, возникает методическая проблема: с одной стороны, необходимо обеспечить освоение базовых языковых средств, с другой — сформировать у школьников системное видение их применения. Путь к преодолению этого противоречия мы видим в перестройке логики учебного занятия, а именно в переходе от разрозненных заданий к совместному созданию программного продукта, где результат каждого участника обретает значение для общего итога.

Материалы и методы исследования

Экспериментальная работа проводилась на базе Частного образовательного учреждения «Газпром школа Санкт-Петербург» в 2025/2026 учебном году. В ней приняли участие 11 обучающихся 10-го класса.

Методологическую основу нашего подхода составили три взаимодополняющих направления: практико-ориентированное, деятельностное и проектное. Практическая направленность предполагает, что программные конструкции изучаются не в отрыве от реальных задач, а в процессе их применения для создания действующих компонентов. Как справедливо отмечает Златопольский, именно контекстное обучение позволяет учащимся глубже осознать функциональное назначение языковых средств и их место в структуре программы [3]. Деятельностный аспект реализуется через прохождение школьниками всех стадий разработки - от первоначального замысла до итоговой проверки и сдачи. Проектный характер обеспечивается наличием конечного продукта и распределением функций между участниками. Лебедева и Тур в своих исследованиях подчёркивают, что проектная деятельность в области информатики даёт наибольший эффект, когда её результатом становится реально работающий продукт [8].

В рамках предлагаемой модели мы трансформируем привычную последовательность «учитель - упражнение - проверка» в иную логику: «проблемная ситуация - проектирование - индивидуальная реализация - тестирование - интеграция - общий продукт». По наблюдениям Вишнякова и его коллег, подобная перестройка способствует выходу обучения на уровень осмысленного конструирования, где каждый этап несёт самостоятельную дидактическую нагрузку [6].

Для сбора эмпирических данных мы использовали методы педагогического наблюдения, анализа созданных программных кодов, анкетирования и рефлексивных опросов.

Результаты исследования

Практическое занятие было организовано в рамках стандартного урока продолжительностью 40 минут. На этапе мотивации мы предложили учащимся проблемную ситуацию: имеется готовый калькулятор, выполняющий базовые арифметические операции; требуется расширить его функциональность за счёт добавления вычисления процентов, квадратного корня, тригонометрических функций и других математических действий, при этом не переписывая существующую программу полностью. Постановка этого вопроса позволила актуализировать ранее усвоенные знания и подвести школьников к самостоятельному выводу о необходимости использования подпрограмм как средства модульной организации кода. В ходе коллективного обсуждения была сформулирована тема занятия. Затем мы кратко повторили теоретический материал, касающийся синтаксиса определения функций в Python, назначения параметров и оператора return.

В качестве основы для практической работы был подготовлен проект «Калькулятор», состоящий из главного исполняемого файла и отдельного модуля, содержащего заготовки подпрограмм. Пример такой заготовки: def square(x): return [9]. Учащимся предстояло дописать тело каждой функции, провести её тестирование и передать готовый код. Список функций, распределённых между участниками, приведён в таблице 1. Каждый обучающийся отвечал за реализацию одной функции, при этом несколько человек объединялись в команды. Такая организация позволила реализовать принцип «индивидуальное поручение — коллективная ответственность — общий результат».

Таблица 1 — Перечень функций для индивидуальной реализации

№	Функция	Имя функции в коде
1	Изменение знака	change_sign(x)
2	Квадрат числа	square(x)

№	Функция	Имя функции в коде
3	Квадратный корень	square_root(x)
4	Синус	sin(x)
5	Тангенс	tan(x)
6	Косинус	cos(x)
7	Два в степени x	two_power(x)
8	Натуральный логарифм	log(x)
9	Возведение в степень	power(x, n)
10	Вычисление процента	percent(x, p)
11	Обратное число	reciprocal(x)

Учащимся был предложен следующий порядок действий: написать код функции, проверить его на нескольких наборах данных (включая граничные и заведомо некорректные значения), отправить результат через электронную форму, при наличии времени выполнить дополнительное задание и по возможности оказать помощь членам команды. Для сбора разработанных фрагментов мы использовали веб-форму, в которой участник указывал фамилию, имя, номер команды, название функции и текст программы. После получения материалов мы проводили проверку и осуществляли интеграцию компонентов в общий проект.

Задания дифференцировались по трём уровням сложности. Базовый уровень предполагал простую реализацию функции в соответствии с поставленной задачей. Повышенный уровень включал добавление проверки ошибочных ситуаций — например, при написании функции деления необходимо было предусмотреть обработку случая деления на ноль. Творческий уровень заключался в разработке дополнительной функции по собственному выбору учащегося. Оценивание осуществлялось по системе

критериев, охватывающих как предметные, так и метапредметные результаты (таблица 2).

Таблица 2 — Критерии оценивания практической работы

Критерий	Содержание
Корректность работы	Функция выдаёт правильный результат
Наличие тестирования	Проверка выполнена на различных входных данных
Обработка исключений	Предусмотрены реакции на некорректные данные
Самостоятельность	Задание выполнено без обращения за помощью
Командная активность	Учащийся включён в совместную работу
Дополнительная активность	Выполнено сверхнормативное задание или оказана помощь

На заключительном этапе мы применили методику рефлексивной самооценки «Лестница успеха»[10]. Учащимся предлагалось определить своё положение на одной из четырёх ступеней: от «понял теорию, но на практике испытываю трудности» до «выполнил задание и могу объяснить решение другим». Такая форма рефлексии, на наш взгляд, позволяет оценить не столько эмоциональное впечатление от урока, сколько степень самостоятельности и уровень освоения материала.

Количественные результаты, полученные в ходе работы, обобщены в таблице 3. Все участники (100%) справились с основным заданием и передали код для сборки. В итоговую версию калькулятора вошли все 11 функций, что обеспечило получение полноценного многофункционального инструмента. Девять школьников (81,8%) провели тестирование своих функций на нескольких наборах данных, включая граничные значения. Шесть учащихся (54,5%) выполнили дополнительные задания повышенного или творческого уровня. Пятеро обучающихся (45,4%) оказали помощь одноклассникам.

Таблица 3 — Результаты выполнения практической работы

Показатель	Количество	Доля, %
Выполнили основное задание	11	100
Провели тестирование	9	81,8
Отправили код через форму	11	100
Выполнили дополнительное задание	6	54,5
Оказали помощь товарищам	5	45,4

Источник: данные педагогического наблюдения и электронного сбора работ автора.

Соотношение этапов деятельности и формируемых компетенций представлено в таблице 4.

Таблица 4 — Связь этапов занятия с формируемыми компетенциями

Этап	Развиваемые умения
Осмысление проблемы	Целеполагание, выделение задачи
Изучение готового кода	Чтение и понимание программ
Написание функции	Применение теоретических знаний
Тестирование	Самоконтроль, корректировка
Совместная работа	Коммуникация, распределение обязанностей
Передача кода	Ответственность, дисциплина
Интеграция	Понимание системной организации
Рефлексия	Самоанализ, оценка вклада

Обсуждение

Систематизация полученных материалов позволяет нам выделить несколько существенных преимуществ предложенной модели. Первое из них

— усиление практической значимости изучаемого содержания. Учащийся наблюдает непосредственное применение созданной им функции в работающем приложении, что существенно повышает внутреннюю мотивацию и осознание ценности собственного труда. Этот вывод согласуется с наблюдениями Новиковой, которая также фиксировала рост познавательного интереса при включении школьников в проектную деятельность [5].

Второе преимущество — персонализация ответственности. Закрепление за каждым участником конкретного программного компонента делает его вклад очевидным и осязаемым как для самого школьника, так и для остальных членов команды.

Третье — гармоничное сочетание индивидуальной и коллективной работы. Обучающийся самостоятельно решает свою задачу, но при этом включён в систему взаимопомощи и взаимодействия, что создаёт благоприятные условия для становления коммуникативных навыков.

Четвёртое — формирование культуры тестирования. Обязательная проверка кода перед отправкой способствует пониманию того, что создание программы не ограничивается написанием исходного текста, а включает этапы верификации, отладки и обработки ошибок.

Пятое — складывание представления о модульной структуре программы. Школьник видит, как его функция встраивается в общую архитектуру, что закладывает основу для последующего знакомства с объектно-ориентированным подходом и принципами модульного проектирования.

Вместе с тем следует указать на ограничения, выявленные в процессе апробации. Данный формат требует от учителя серьёзной предварительной подготовки: необходимо разработать и отладить базовый проект, определить перечень функций, подготовить заготовки и тестовые данные, сформулировать критерии оценки. При значительной наполняемости классов ручная интеграция компонентов может стать трудоёмкой, что делает

перспективным использование автоматизированных систем проверки и контроля версий.

Заключение

Проведённое исследование подтверждает состоятельность практико-ориентированной методики обучения программированию, построенной на создании коллективного программного продукта. Разработанная модель организации занятия по теме «Разработка подпрограмм» позволяет преодолеть разрыв между формальным усвоением синтаксических конструкций и пониманием их функционального назначения.

Ключевой результат нашей работы — методически обоснованная и апробированная модель, охватывающая последовательность этапов от проблемной ситуации до рефлексивной оценки, дифференциацию заданий по уровням сложности, систему критериев и цифровые каналы сбора результатов. Эмпирические данные свидетельствуют о достижении поставленных целей: все участники выполнили базовое задание, большинство осуществили тестирование разработанных функций, более половины проявили активность в выполнении дополнительных задач.

Предлагаемая методика позволяет одновременно решать предметные задачи (формирование навыков программирования, отладки и проверки кода) и метапредметные задачи (развитие умений планирования, самоконтроля, коммуникации и рефлексии). Изучение подпрограмм выводится за рамки отдельного упражнения — функция начинает восприниматься как самостоятельный программный компонент, интегрированный в общую систему.

Практическая значимость результатов заключается в возможности тиражирования предложенной модели в общеобразовательных организациях без дополнительного программного обеспечения и временных затрат, а также в её адаптации к другим темам школьного курса информатики.

Перспективными направлениями дальнейшей работы мы считаем адаптацию предложенной модели к другим разделам курса информатики

(обработка файлов, структуры данных, классы и объекты), а также расширенную экспериментальную проверку её эффективности на большей выборке. Разработанный подход может рассматриваться как одна из результативных форм практико-ориентированного преподавания программирования в старшей школе, обладающая значительным потенциалом для распространения в педагогической практике.

Литература

1. Федеральный государственный образовательный стандарт среднего общего образования: утверждён приказом Министерства образования и науки Российской Федерации от 17.05.2012 № 413 (с изменениями, внесёнными приказом Министерства просвещения Российской Федерации от 12.08.2022 № 732). — Текст: электронный.

2. Босова Л. Л., Босова А. Ю. Информатика: учебник для 10 класса (базовый уровень). — 8-е изд., стер. — Москва: Просвещение, 2025. — 288 с. — ISBN 978-5-09-127116-4. — URL: https://ibooks.ru/bookshelf/401486/reading (дата обращения: 26.08.2026). — Текст: электронный.

3. Златопольский Д. М. Программирование: типовые задачи, алгоритмы, методы: учебное пособие. — 4-е изд. — Москва: Лаборатория знаний, 2020. — 224 с. — ISBN 978-5-00101-789-9.

4. Златопольский Д. М. Программирование: типовые задачи, алгоритмы, методы. — 5-е изд., электрон. — Москва: Лаборатория знаний, 2025. — 226 с. — ISBN 978-5-93208-832-6. — URL: https://start.rucont.ru/efd/443549 (дата обращения: 26.08.2026). — Текст: электронный.

5. Новикова Е. И. Использование метода проекта на уроке информатики с целью повышения мотивации школьников // Сибирский учитель. — 2010. — № 5. — С. 24–29. — URL: https://www.sibuch.ru/taxonomy/term/383 (дата обращения: 26.08.2026). — Текст: электронный.

6. Вишняков С. В., Лазарев В. И., Вишнякова Ю. Н. Опыт организации проектной деятельности школьников по направлению «Информационные технологии» // Современные информационные технологии и ИТ-образование. — 2024. — Т. 20, № 3. — С. 4–12.

7. Королев А. Л. Проектная инженерная деятельность в школьном образовании // Вестник Шадринского государственного педагогического университета. — 2019. — № 3. — С. 45–52.

8. Лебедева М. Б., Тур С. Н. Организация проектной деятельности при изучении информатики в начальной школе // Ярославский педагогический вестник. — 2011. — № 4. — С. 78–83.

9. Python Software Foundation. The Python Tutorial: Defining Functions [Электронный ресурс]. — Режим доступа: https://docs.python.org/3/tutorial/controlflow.html#defining-functions (дата обращения: 26.08.2026).

10. Спанчок А. В. Программа развития классного коллектива "Лестница успеха" [Электронный ресурс] // Инфоурок. – URL: https://infourok.ru/programma-razvitiya-klassnogo-kollektiva-lestnica-uspeha-1397617.html (дата обращения: 26.08.2026)