Центр цифрового образования детей «IT- куб»
(структурное подразделение АНО ДТ «Красноярский Кванториум»)

РЕКОМЕНДОВАНО
методическим советом

Протокол № 1
от «27» мая 2025 г.

УТВЕРЖДАЮ
Генеральный директор
АНО ДТ «Красноярский Кванториум»
_____ Кениг С.Р.
Приказ № IT-11
от «30» мая 2025 г.

Дополнительная общеобразовательная общеразвивающая программа
технической направленности

«Основы программирования на языке Python»

Срок реализации:
1 год
Возраст детей:
11-12 лет
Составитель программы:
Бухарова Т.В.

г. Красноярск, 2025 г.

1. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Данная программа разработана в соответствии с нормативными правовыми актами в области образования:

Федеральным законом от 29 декабря 2012 № 273-ФЗ «Об образовании в Российской Федерации»;

Указом Президента РФ от 07.05.2018 № 204 «О национальных целях и стратегических задачах развития Российской Федерации на период до 2024 года»;

Распоряжением Правительства РФ от 29.05.2015 № 996-р «Стратегия развития воспитания в Российской Федерации до 2025 года, утвержденная»;

Распоряжением Правительства РФ от 31.03.2022 № 678-р «Концепция развития дополнительного образования детей до 2030»;

Постановлением Правительства Российской Федерации от 26.12.2017 № 1642 «Об утверждении государственной программы Российской Федерации «Развитие образования»;

Приказом Министерства просвещения РФ от 27.07.2022 №629 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам»;

Постановлением Главного государственного санитарного врача Российской Федерации от 28.09.2020 № 28 «Об утверждении санитарных правил СП 2.4.3648-20 «Санитарно-эпидемиологические требования к организациям воспитания и обучения, отдыха и оздоровления детей и молодежи»;

Методическими рекомендациями по проектированию дополнительных общеразвивающих программ (включая разноуровневые программы) (Приложение к письму Министерства образования и науки Российской Федерации от 18.11.2015 г. № 09-3242);

Методическими рекомендациями по реализации дополнительных общеобразовательных программ с применением электронного обучения и дистанционных образовательных технологий (Приложение к письму Министерства просвещения РФ от 31.01.2022 г. № 1ДГ 245/06);

Методическими рекомендациями «Об использовании государственных символов Российской Федерации при обучении и воспитании детей и молодежи в образовательных организациях, а также организациях отдыха детей и их оздоровления» (Письмо Министерства просвещения РФ от 15.04.2022 № СК-295/06);

Методическими рекомендациями по созданию и функционированию центров цифрового образования «IT-куб» (Письмо Министерства просвещения РФ от 10.11.2021 №ТВ-1984/04.

Дополнительная общеобразовательная общеразвивающая программа «Основы программирования на языке Python» (далее - программа) имеет техническую направленность, стартовый уровень сложности и

ориентирована на обучающихся 11-12 лет. Программа в объеме 144 часа рассчитана на один год из расчета 4 часа в неделю.

1.1. Актуальность программы

Актуальность программы "Программирование на языке Python для детей 11-12 лет" чрезвычайно высока, и вот почему:

1. Востребованность на рынке труда:
   - Цифровизация: В современном мире цифровые технологии проникают во все сферы жизни, и знание программирования становится важным навыком.
   - Дефицит IT-специалистов: На рынке труда наблюдается высокий спрос на квалифицированных специалистов в области IT, включая программистов.
   - Перспективы карьерного роста: Обучение программированию открывает перед детьми широкие перспективы карьерного роста в будущем, в том числе в областях, которые еще только формируются.
2. Развитие когнитивных навыков:
   - Логическое мышление: Программирование развивает логическое мышление, способность анализировать задачи и разбивать их на более мелкие части.
   - Абстрактное мышление: Дети учатся абстрагироваться от конкретных деталей и мыслить в терминах общих алгоритмов.
   - Алгоритмическое мышление: Программирование учит строить алгоритмы - последовательности действий для решения задач.
   - Критическое мышление: Программисты учатся находить и исправлять ошибки в своем коде, что развивает критическое мышление и навыки решения проблем.
   - Пространственное мышление: При работе с графикой и анимацией (например, в играх) развивается пространственное воображение.
3. Улучшение образовательных результатов:
   - Поддержка других предметов: Знание программирования может помочь детям лучше понимать математику, физику, информатику и другие предметы.
   - Креативность и самовыражение: Программирование позволяет детям создавать собственные игры, анимации, веб-сайты и другие проекты, что развивает их творческие способности и помогает им самовыражаться.
4. Удобство и доступность языка Python:
   - Простота изучения: Python является одним из самых простых и понятных языков программирования для начинающих. Его синтаксис прост и интуитивно понятен.

- Многофункциональность: Python можно использовать для решения широкого спектра задач, от разработки игр и веб-приложений до анализа данных и машинного обучения.
- Большое сообщество: Python имеет большое и активное сообщество разработчиков, которое предоставляет поддержку, документацию и множество готовых решений.
- Открытый код: Python является языком с открытым кодом, что означает, что его можно использовать бесплатно и распространять свободно.
- Наличие библиотек: Python имеет огромный набор библиотек (готовых инструментов) для различных задач, что упрощает процесс разработки. Например, библиотеки для создания игр (Pygame), работы с графикой (Matplotlib), обработки данных (Pandas).

5. Создание цифрового будущего:
   - Развитие цифровой грамотности: Программа способствует развитию цифровой грамотности у детей, что является важным навыком в современном мире.
   - Подготовка к будущему: Обучение программированию позволяет детям подготовиться к будущим профессиям, связанным с технологиями.
   - Участие в создании технологий: Дети получают возможность создавать новые технологии и продукты, а не только ими пользоваться.
6. Мотивация и интерес:
   - Игровой формат: Программирование на Python можно преподавать в игровой форме, что делает процесс обучения более интересным и увлекательным для детей.
   - Наглядные результаты: Дети быстро видят результаты своей работы - созданные игры, анимации, сайты - что повышает их мотивацию и интерес к обучению.
   - Удовлетворение от творчества: Программирование позволяет детям реализовать свои творческие идеи и создавать что-то новое.

1.2. Отличительные особенности:

Программа авторская, разработана на основе методического пособия «Реализация дополнительной общеобразовательной программы по тематическому направлению «Основы программирования на языке Python» с использованием оборудования центра цифрового образования детей «IT-куб»», г. Москва, 2021 г.

1. Адаптированный контент и методика:
   - Учет возрастных особенностей: Программа специально разработана для детей 11-12 лет, учитывая их когнитивные способности, интересы и уровень концентрации.

- Простота и наглядность: Материал излагается простым и понятным языком, с использованием наглядных примеров, иллюстраций и аналогий.
- Игровой подход: Обучение проводится в игровой форме, с использованием интерактивных упражнений, головоломок и задач, что делает процесс более увлекательным и мотивирующим.
- Практическая направленность: Основной упор делается на практическое применение знаний, создание реальных проектов (игр, анимаций, веб-сайтов и т.д.).
- Небольшие проекты: Вместо крупных, сложных проектов, программа предлагает выполнение небольших, управляемых проектов, которые позволяют детям быстро увидеть результаты своей работы.

2. Фокус на Python как на языке программирования:

- Простота Python: Выбор Python обусловлен его простотой, читабельностью и интуитивно понятным синтаксисом, что делает его идеальным языком для начинающих программистов.
- Широкие возможности Python: Подчеркивается универсальность Python и его применимость в различных областях (веб-разработка, анализ данных, машинное обучение, игровая разработка).
- Акцент на библиотеках для детей: Программа может использовать специальные библиотеки, разработанные для детей и начинающих, такие как Turtle (графика), Pygame (игры) или Arcade (игры). Это позволяет быстро начать создавать интересные проекты.

1.3. Адресат программы, требования к обучающимся, возрастные особенности группы.

Набор на программу осуществляется в соответствии с Порядком приема и отчисления обучающихся автономной некоммерческой организации «Красноярский детский технопарк «Кванториум».

Возраст обучающихся:

Программа «Основы программирования на языке Python» рассчитана на обучающихся 11-12 лет. В связи с ориентированностью программы на практическую индивидуальную (групповую) работу максимальное количество обучающихся в группе не должно превышать 8 человек.

1.4. Педагогическая целесообразность

Программа «Основы программирования на языке Python»:

- обеспечивает знакомство с фундаментальными понятиями алгоритмизации и программирования на доступном уровне;
- имеет практическую направленность с ориентацией на реальные потребности, соответствующие возрасту ученика;
- охватывает как алгоритмическое направление, так и вопросы практического использования полученных знаний при решении задач из различных областей знаний;

- ориентирована на существующий парк вычислительной техники и дополнительные ограничения;
- допускает возможность варьирования в зависимости от уровня подготовки и интеллектуального уровня учащихся (как группового, так и индивидуального);
- предусматривает возможность индивидуальной работы с учащимися.

Практическая значимость программы заключается в том, что он способствует более успешному овладению знаниями и умениями по направлению «Программирование» через развитие самостоятельности обучающихся и оптимизацию средств и методов обучения.

1.5. Срок реализации программы и объём учебных часов

Программа рассчитана на 1 год обучения. Годовая нагрузка на обучающегося составляет 144 часа.

1.6. Режим занятий, формы и методы обучения

Учебные занятия проходят в очной форме. Режим занятий – 2 раза в неделю по 2 академических часа (1 академический час - 40 минут) с обязательным перерывом, что определяется Санитарно-эпидемиологическими правилами и нормативами СанПиН 2.4.3648-20.

При проведении занятий используются комбинированные занятия – изложение нового материала, проверка пройденного материала, закрепление полученных знаний, самостоятельная работа.

При проведении занятий используются три формы работы:

- демонстрационная, когда обучающиеся слушают объяснения педагога и наблюдают за демонстрационным экраном или экранами компьютеров на ученических рабочих местах;
- фронтальная, когда обучающиеся синхронно работают под управлением педагога;
- самостоятельная, когда обучающиеся выполняют индивидуальные задания в течение части занятия.

Повторение и усвоение пройденного материала осуществляется через прикладную работу обучающегося, использующего на практике приобретенные знания.

1.7. Цель и задачи программы

Целью программы «Основы программирования на языке Python» является создание условий для изучения методов программирования на языке Python; рассмотрение различных парадигм программирования, предлагаемых этим языком (процедурная, функциональная, объектно-ориентированная); подготовка к использованию как языка программирования, так и методов программирования на Python в учебной и последующей профессиональной деятельности в различных предметных областях.

Задачи:

- формирование и развитие навыков алгоритмического и логического мышления, грамотной разработки программ;
- знакомство с принципами и методами функционального программирования;
- знакомство с принципами и методами объектно-ориентированного программирования; • приобретение навыков работы в интегрированной среде разработки на языке Python;
- изучение конструкций языка программирования Python;
- знакомство с основными структурами данных и типовыми методами обработки этих структур;
- приобретение навыков разработки эффективных алгоритмов и программ на основе изучения языка программирования Python;
- приобретение навыков поиска информации в сети Интернет, анализ выбранной информации на соответствие запросу, использование информации при решении задач;
- развитие у обучающихся интереса к программированию;
- формирование самостоятельности и творческого подхода к решению задач с использованием средств вычислительной техники;
- воспитание упорства в достижении результата;
- расширение кругозора обучающихся в области программирования.

1.8. Планируемые результаты освоения программы

По результатам обучения, обучающиеся приобретает следующие компетенции:

- знание основ современных языков программирования;
- умение объяснять и использовать на практике как простые, так и сложные структуры данных и конструкции для работы с ними;
- умение искать и обрабатывать ошибки в коде;
- умение разбивать решение задачи на подзадачи;
- способность писать грамотный, красивый код;
- способность анализировать как свой, так и чужой код;
- способность работать с информацией: находить, оценивать и использовать информацию из различных источников, необходимую для решения профессиональных задач (в том числе на основе системного подхода); способность грамотно строить коммуникацию, исходя из целей и ситуации.

1.9. Формы подведения итогов обучения

При организации занятий по программе «Основы программирования на языке Python» для достижения поставленных целей и решения поставленных задач используются формы проведения занятий с активными методами обучения:

- занятие в форме проблемно-поисковой деятельности;

- занятие с использованием межпредметных связей;
- занятие в форме мозгового штурма;
- занятие в форме частично-поисковой деятельности.

Формы и методы контроля:

- тестирование;
- устный опрос;
- самостоятельные и контрольные работы;
- участие в проектной деятельности.

Общая характеристика учебного процесса:

- при изучении программы используются практические и самостоятельные работы;
- программа обучения заканчивается написанием программы для решения одной из задач.

2. УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН

№	Наименование раздела, модуля	Количество академических часов			Форма контроля
		Всего	Теория	Практика	
1	**Знакомство с направлением обучения**	**2**	**1**	**1**	
1.1	Вводный урок. Правила и техника безопасности при работе с оборудованием.	2	1	1	Беседа
2	**Модуль 1. Введение в программирование**	**28**	**7**	**21**	
2.1	Знакомство с программой и с LMS. Установка и настройка среды программирования. PEP 8. Ввод-вывод. Переменные	4	1	3	Практическое задание
2.2	Условный оператор. Отступы. Операции над строками	4	1	3	Практическое задание
2.3	Сложные условия. Вложенные структуры	4	1	3	Практическое задание
2.4	Типы данных. Операции над числами	4	1	3	Практическое задание
2.5	Приоритет операций. Простейшие функции	4	1	3	Практическое задание
2.6	Пробная самостоятельная работа	4	1	3	Практическое задание
2.7	Самостоятельная работа	4	1	3	Практическое задание
3	**Модуль 2. Базовые конструкции языка Python 3**	**74**	**20**	**54**	
3.1	Цикл с предусловием	8	4	4	Практическое задание
3.2	Цикл с параметром	8	4	4	Практическое задание
3.3	Самостоятельная работа	4		4	Практическое задание
3.4	Булевы переменные. Прерывания и продолжения циклов	4	1	3	Практическое задание
3.5	Вложенные циклы	4	1	3	Практическое задание
3.6	Контрольная работа	4		4	Практическое задание
3.7	Элементы теории множеств. Множества в Python	4	1	3	Практическое задание
3.8	Строки. Индексация	4	1	3	Практическое задание

3.9	Строки. Срезы	4	1	3	Практическое задание
3.10	Самостоятельная работа	4		4	Практическое задание
3.11	Знакомство со списками	4	1	3	Практическое задание
3.12	Кортежи	4	1	3	Практическое задание
3.13	Списочные выражения. Методы split() и join()	4	1	3	Практическое задание
3.14	Другие методы списков и строк	4	1	3	Практическое задание
3.15	Самостоятельная работа	2		2	Практическое задание
3.16	Вложенные списки	2	1	1	Практическое задание
3.17	Знакомство со словарями	2	1	1	Практическое задание
3.18	Методы словарей	2	1	1	Практическое задание
3.19	Контрольная работа	2		2	Практическое задание
4	**Библиотеки Tkinter, Pygame**	**20**	**8**	**12**	
4.1	Знакомство с библиотекой Tkinter	10	4	6	Практическое задание
4.2	Знакомство с библиотекой Pygame	10	4	6	Практическое задание
5	**Итоговая аттестация.**	**16**	**4**	**12**	
8.1	Проектная работа	14	4	10	Творческий проект
8.2	Итоговая аттестация.	2	0	2	Кейсовое задание
Итог		**144**	**26**	**46**	

3. СОДЕРЖАНИЕ ПРОГРАММЫ

1. Знакомство с направлением обучения.

1.1 Вводный урок. Правила и техника безопасности при работе с оборудованием.

Теоретическая работа: вводная лекция на тему «История языков программирования».

2. Модуль 1. Введение в программирование

2.1. Знакомство с программой и с LMS. Установка и настройка среды программирования. PEP 8. Ввод-вывод. Переменные

Теоретическая работа: Знакомство с виртуальной средой взаимодействия: регистрация, организация личного кабинета, поиск и выкладывание материалов. Знакомство с системой автоматизированной проверки задач. Основные понятия программирования: исполнитель, система команд, алгоритм, программа, среда разработки, интерпретатор, код программы и редактор кода.

Практическая работа: решение задач с функциями ввод и вывод

2.2. Условный оператор. Отступы. Операции над строками

Теоретическая работа: Простейшие программы с использованием условного оператора if

Практическая работа: решение задач с условным оператором

2.3. Сложные условия. Вложенные структуры

Теоретическая работа: Простейшие программы с использованием условного оператора elif

Практическая работа: решение задач с условным оператором

2.4. Типы данных. Операции над числами

Теоретическая работа: Скалярные типы данных

Практическая работа: решение задач со строками и числами

2.5. Приоритет операций. Простейшие функции

Теоретическая работа: Длина строки, максимум и минимум

Практическая работа: решение задач

2.6. Пробная самостоятельная работа

Теоретическая работа: Функция округление

Практическая работа: решение задач

2.6. Самостоятельная работа

Практическая работа: решение задач

3. Модуль 2. Базовые конструкции языка Python 3

3.1. Цикл с предусловием

Теоретическая работа: Цикл while

Практическая работа: решение задач с использованием цикла

3.2. Цикл с параметром

Теоретическая работа: цикл for

Практическая работа: решение задач с использованием цикла.

3.3. Самостоятельная работа

Практическая работа: решение задач с использованием цикла.

3.4. Булевы переменные. Прерывания и продолжения циклов

Теоретическая работа: конструкция break

Практическая работа: решение задач с использованием циклов.

3.5. Вложенные циклы
Теоретическая работа: Оператор break и continue во вложенных циклах
Практическая работа: решение задач с использованием циклов.
3.6. Контрольная работа
Практическая работа: решение задач с использованием циклов.
3.7. Элементы теории множеств. Множества в Python
Теоретическая работа: Теория множества, создание множества, операции над множеством
Практическая работа: решение задач
3.8. Строки. Индексация
Теоретическая работа: Перебор элементов строки
Практическая работа: решение задач со строками
3.9. Строки. Срезы
Теоретическая работа: Срез строки
Практическая работа: решение задач со строками
3.10. Самостоятельная работа
Практическая работа: решение задач со строками
3.11. Знакомство со списками
Теоретическая работа: Создание списка, добавление, удаление и замена элемента в списке
Практическая работа: решение задач со списками
3.12. Кортежи
Теоретическая работа: Сортировка пузырьком
Практическая работа: решение задач с кортежами
3.13. Списочные выражения. Методы split() и join()
Теоретическая работа: Объединение строк
Практическая работа: решение задач
3.14. Списочные выражения. Методы split() и join()
Теоретическая работа: Объединение строк
Практическая работа: решение задач
3.15. Другие методы списков и строк
Теоретическая работа: Методы dir и help
Практическая работа: решение задач
3.16. Вложенные списки
Теоретическая работа: Методы Создание двумерного массива
Практическая работа: решение задач
3.17. Знакомство со словарями
Теоретическая работа: Создание словаря. Добавление и удаление элемента словаря
Практическая работа: решение задач со словарями
3.18. Методы словарей
Теоретическая работа: Допустимые типы ключей
Практическая работа: решение задач со словарями

3.19. Контрольная работа
Практическая работа: решение задач со словарями

4. Модуль 3. Знакомство с библиотеками

4.1. Библиотека Tkinter
Теоретическая работа: основные виджиты окна
Практическая работа: решение задач
4.2. Библиотека Pygame
Теоретическая работа: отрисовка геометрических фигур
Практическая работа: решение задач

5. Итоговая аттестация.

5.1. Проектная работа – творческий проект
5.2 Практическая работа: презентация кейсового задания по программированию

СПИСОК ЛИТЕРАТУРЫ

1. К. Ю. Поляков, Е. А. Еремин. Информатика. Углублённый уровень. Учебник для 10 класса в 2 частях. М.: БИНОМ. Лаборатория знаний, 2014.
2. М. Лутц. Изучаем Python. СПб.: Символ-Плюс, 2011.
3. Задачи по программированию. Под ред. С. М. Окулова, М.: БИНОМ. Лаборатория знаний, 2006.
4. С. М. Окулов. Основы программирования. М.: Бином. Лаборатория знаний, 2012.

Литература, рекомендованная обучающимся

1. М. Лутц. Изучаем Python. СПб.: Символ-Плюс, 2011.
2. Информатика и ИКТ. Задачник-практикум в 2 частях. Под ред. И. Г. Семакина и Е. К. Хеннера. М.: БИНОМ. Лаборатория знаний, 2014.

Ресурсы в интернете

1. Материалы и презентации к урокам в LMS Яндекс.Лицея.
2. Сайт pythonworld.ru — «Python 3 для начинающих».
3. Сайт pythontutor.ru — «Питонтьютор».
4.https://www.youtube.com/playlist?list=PLJOzdkh8T5kpIBTG9mM2wVBjh5OpdwBl — Лекции А.В. Умнова, прочитанные в Школе Анализа Данных Яндекса.

5. ИНФОРМАЦИОННО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ И МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОСНАЩЕНИЕ ПРОГРАММЫ

5.1. Материально-техническое обеспечение программы

№ п/п	Наименование оборудования (ФПО)	Примерная модель (РВПО)	Единица измерения	Количество
1	**"Компьютерное оборудование"**			
1.1	МФУ (принтер, сканер, копир)	Epson L14150	шт	1
1.2	Ноутбук Тип 3	Lenovo Legion Y545 (81Q6000URU)	шт	14
2	**"Презентационное оборудование"**			
2.1	Напольная мобильная стойка для интерактивных досок или универсальное настенное крепление	ONKRON TS1330	шт	1
2.2	Моноблочное интерактивное устройство	SMART SBID-MX265-V2	шт	1
3	**"Дополнительное оборудование"**			
3.1	Комплект комплектующих и расходных материалов	Xiaomi Wiha Precision Screwdriver (DZN4002TY)	шт	4
3.2	Комплект кабелей и переходников	Atcom High speed HDMI - HDMI MOST Lite LRG ФАZA FOP-05GS-500	шт	1
3.3	Флипчарт магнитно-маркерный на треноге	Attache	шт	1

5.2. Информационное обеспечение программы

Программное обеспечение python IDE 3.10

Приложение. Практические работы

1. Угадай число

Эта игра поможет детям понять основы условных операторов и циклов.

```
import random

number_to_guess = random.randint(1, 100)

while True:
    guess = int(input("Угадай число от 1 до 100: "))

    if guess == number_to_guess:
        print("Поздравляем, вы угадали число!")
        break
    elif guess < number_to_guess:
        print("Загаданное число больше.")
    else:
        print("Загаданное число меньше.")
```

2. Калькулятор

Напишите простую программу-калькулятор, которая будет выполнять основные математические операции: сложение, вычитание, умножение и деление.

```
def calculator():
    operation = input("Выберите операцию (+, -, *, /): ")
    num1 = float(input("Введите первое число: "))
    num2 = float(input("Введите второе число: "))

    if operation == '+':
        result = num1 + num2
    elif operation == '-':
        result = num1 - num2
    elif operation == '*':
        result = num1 * num2
    elif operation == '/':
        result = num1 / num2
    else:
        print("Некорректная операция.")
        return
```

```
    print("Результат:", result)

calculator()
```

3. Генератор случайных фактов

Напишите программу, которая будет выводить случайные факты на экран. Создайте список интересных фактов и используйте функцию random.choice(), чтобы выбирать случайный факт из списка.

```
import random

facts = [
    "Кошки спят в среднем 13-14 часов в день.",
    "Слоны могут слышать звуки на расстоянии до 4 километров.",
    "Человеческое сердце создает достаточно давления, чтобы выбросить кровь на высоту до 10 метров."
]

random_fact = random.choice(facts)
print(random_fact)
```

4. Словарь терминов

Создайте программу, которая будет служить словарём терминов по программированию.

```
programming_dictionary = {
    "переменная": "именованное хранилище данных",
    "цикл": "конструкция, повторяющая выполнение определенного блока кода",
    "функция": "блок кода, предназначенный для выполнения определенной задачи"
}

term = input("Введите термин для определения: ")

if term in programming_dictionary:
    print(programming_dictionary[term])
    print("Термин не найден в словаре.")
```

5. Викторина

Создайте список вопросов и ответов, затем используйте циклы и условные операторы для проверки правильности ответа.

```
questions = {
   "Как называется столица Франции?": "Париж",
   "Сколько планет в Солнечной системе?": "Восемь",
   "Как называется жидкость, заполняющая глазное яблоко?": "Слеза"
}

score = 0

for question, answer in questions.items():
   user_answer = input(question)

   if user_answer.lower() == answer.lower():
      print("Правильно!")
      score += 1
   else:
      print("Неправильно. Правильный ответ:", answer)

print("Итоговый счет:", score)
```

6. Проверка на чётность

На всякий случай напомним, что чётное число делится на 2, а нечётное соответственно — не делится. В этой задаче мы напишем функцию check_even_odd, которая принимает число и проверяет, является ли оно чётным или нечётным.

```
num = int(input("Введите число: "))
if num % 2 == 0:
   print("Число", num, "четное")
else:
   print("Число", num, "нечетное")
```

7. Подсчёт суммы чисел в списке

```
numbers = [10, 20, 30, 40, 50] # Пример списка чисел
```

```
# Используем встроенную функцию sum() для подсчета суммы чисел в списке
total = sum(numbers)

print("Сумма чисел в списке:", total)
```

8. Проверка на палиндром
Палиндром — это слово, которое читается одинаково в обоих направлениях

```
def is_palindrome(word):
    word = ''.join(e for e in word if e.isalnum()).lower()
    return word == word[::-1]

word_to_check = "А роза упала на лапу Азора"
if is_palindrome(word_to_check):
    print(word_to_check + " - это палиндром!")
else:
    print(word_to_check + " - это не палиндром.")
```

9. Поиск максимального числа в списке

```
numbers = [10, 30, 20, 50, 40] # Пример списка чисел

# Используем встроенную функцию max() для поиска максимального числа в списке
max_number = max(numbers)

print("Максимальное число в списке:", max_number)
```

10. Вывести таблицу умножения

```
num = 5
for i in range(1, 11):
    print(num, "x", i, "=", num * i)
```