УТВЕРЖДАЮ
Директор НОЧУ ДПО УЦ «Сетевая Академия»

_____/Соколова Н.М./

Негосударственное образовательное частное учреждение
дополнительного профессионального образования
Учебный центр "Сетевая Академия"

Дополнительная общеобразовательная
(общеразвивающая) программа

«Программирование на языке Python: путь от первых проектов к реальным проектам (базовый уровень)»

Категория учащихся: учащиеся 9-11 классов средней школы
Срок освоения программы: 144 часа

Авторы программы:
Гуляев Е.Н.

Москва, 2024 год

Оглавление

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Учебный курс «Программирование на языке Python: путь от первых проектов к реальным проектам (базовый уровень)»» является частью образовательной программы для учеников средней и старшой школы.

Предлагаемая программа соответствует положениям федерального государственного образовательного стандарта среднего общего образования.

Программа курса отражает способы формирования универсальных учебных действий, составляющих основу для профессионального самоопределения, саморазвития и непрерывного образования, выработки коммуникативных качеств, целостности общекультурного, личностного и познавательного развития учащихся.

Рабочая программа составлена на основе:

- Закона об образования Российской Федерации;
- Федерального государственного образовательного стандарта среднего общего образования;
- Профессиональных стандартов: 06.001 Программист, 06.028 Системный программист, 06.035 Разработчик Web и мультимедийных приложений.

Программа соответствует требованиям к структуре программ, заявленным в ФГОС, и включает следующие разделы:

- Пояснительная записка, в которой уточняются общие цели образования
- с учетом специфики курса.
- Общая характеристика курса, содержащая ценностные ориентиры образования по профилю «Программирование».
- Место данного курса в учебном плане.
- Результаты освоения курса (личностные, метапредметные и предметные), соответствующие глобальным целям образования по профилю «Программирование» и принципу развивающего обучения, лежащему в основе предлагаемой программы.
- Содержание курса по направлению «Программирование на языке Python: путь от первых проектов к реальным проектам (базовый уровень)» в 9 и 11 классах.
- Тематическое планирование, которое дает представление об основных видах учебной деятельности в процессе освоения курса в 9 -11 классах основной школы.
- Рекомендации по учебно-методическому и материально-техническому обеспечению образовательного процесса.

ЦЕЛИ И ЗАДАЧИ КУРСА «ПРОГРАММИРОВАНИЯ»

Целями курса является формирование у обучающегося алгоритмического мышления и, соответственно, необходимых знаний и умений, необходимых для успешного развития в направлении дальнейшей деятельности в области программирования.

Для достижения поставленных целей образование в области разработки программных средств призвано обеспечить решение следующих задач:

- формирование в процессе решения практических задач у учащихся навыков алгоритмического мышления и понимания средств формального описания алгоритмов;
- овладение приёмами написания программ на языках программирования с использованием основных конструкций;
- осознание практической применимости выполняемых учебных задач в современном обществе для возможного выбора этой области в качестве будущей профессии.

ПРИНЦИПЫ И ПОДХОДЫ К ФОРМИРОВАНИЮ ПРОГРАММЫ

Реализация принципа системности, обеспечение связности профиля «программирование» с другими разделами и темами информатики.

Самым важным принципом в процессе обучения программированию является решение практических задач и участие в учебных проектах. В результате освоения успешного программы ученики смогут понять принципы большинства современных языков программирования и применить на практике полученные знания.

СОСТАВ УЧАСТНИКОВ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА

Программа основного общего образования рассчитана на реализацию в 9–11 классах общеобразовательных учреждений и учреждений с углубленным изучением отдельных предметов, и нацелена на возрастную категорию учащихся 14–18 лет.

ОБЩАЯ ХАРАКТЕРИСТИКА УЧЕБНОГО КУРСА

Представленная программа направления «Программирование на языке Python: путь от первых проектов к реальным проектам (базовый уровень)» (9-11 класс) предназначена для практического освоения учащимися следующих видов профессиональной деятельности:

- разработка и отладка программного кода;
- тестирование и рефакторинг программного кода;

- разработка требований и проектирование программного обеспечения;
- интеграция программных модулей и компонент и верификация выпусков программного продукта.

Программа курса «Программирование» составлена на 144 часа.
Курс на 144 часа можно изучать за два учебных года по 72 часа соответственно (4 академических часа в неделю)

При изучении курса 144 часа предполагается углубленное изучение тем, более серьезную отработку навыков за счет расширения практической части, а также выделено больше времени на разработку проектов.

Содержание курса направлено на формирование универсальных учебных действий, обеспечивающих развитие познавательных и коммуникативных качеств личности. Обучающиеся включаются в деятельность, основу которой составляют такие учебные действия, как умение видеть проблемы, ставить вопросы, классифицировать, наблюдать, делать выводы, объяснять, доказывать, защищать свои идеи, давать определение понятиям, структурировать материал и др. обучающиеся включаются в коммуникативную учебную деятельность, где преобладают такие ее виды, как умение полно и точно выражать свои мысли, аргументировать свою точку зрения, работать в группе, представлять и сообщать информацию в устной и письменной форме, вступать в диалог и др.

Программа курса «Программирование на языке Python: путь от первых проектов к реальным проектам (базовый уровень)» для средней и старшей школы предусматривает реализацию следующих принципов:

- придать развитию знаний динамичный характер: использовать ранее полученные знания при овладении новыми понятиями, постепенно углублять и развивать ведущие понятия в процессе изучения всего курса;
- сконцентрировать учебный материал, укрупнив комплектные единицы знаний, что создает дидактические условия для развития системного мышления у учащихся: освободить учебный материал от деталей, имеющих специальное значение, но излишних для общего образования, группируя при этом частные понятия, необходимые для общего образования, вокруг ведущих понятий;
- формировать у обучающихся системное мышление, сочетая его с активной познавательной деятельностью обучающихся;
- учитывать возрастные, индивидуальные особенности и возможности обучающихся, предлагая им задания по выбору.

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОСВОЕНИЯ УЧЕБНОГО КУРСА

Обучение по направлению «Программирование на языке Python: путь от первых проектов к реальным проектам (базовый уровень)» направлено на достижение обучающимися следующих результатов.

Личностные результаты:

- формирование алгоритмического мышления;
- формирование интеллектуальных умений: анализировать информацию, анализировать основные изученные понятия, строить рассуждения, анализировать и сопоставлять теоретические знания с их практической применимостью;
- готовность и способность к самостоятельной, творческой и ответственной деятельности;
- навыки сотрудничества со сверстниками, детьми младшего возраста, взрослыми в образовательной, общественно полезной, учебно-исследовательской, проектной и других видах деятельности;
- готовность и способность к образованию, в том числе самообразованию;
- сознательное отношение к непрерывному образованию как условию успешной профессиональной и общественной деятельности;
- осознанный выбор будущей профессии и возможностей реализации собственных жизненных планов;
- отношение к профессиональной деятельности как возможности участия в решении личных, общественных, государственных, общенациональных проблем;
- понимание роли информационных процессов в современном мире;
- владение первичными навыками анализа и критичной оценки получаемой информации.

Метапредметные результаты:

- знание общепредметных понятий: информация, данные, алгоритм, исполнитель, программа, программирование, языки программирования, система, функция, объект,
- владение универсальными умениями: постановка задачи, формулирование проблемы; поиск, выделение и структурирование необходимой информации; выбор наиболее эффективных методов решения задачи в зависимости от конкретных условий; самостоятельное создание алгоритмов решения задач;
- умение применить изученные понятия для реализации учебных задач;

- умение анализировать имеющийся инструментарий и применять его к поставленной задаче;
- умение анализировать результат своей предыдущей деятельности и приводить его к виду, требуемому на следующем этапе;
- умение работать с различными источниками информации, применять на практике полученные знания, анализировать модели.

Предметные результаты:

- преобразовывать абстрактную идею в последовательность конкретных шагов, необходимых для её воплощения на практике;
- выполнять дискретизацию задачи как необходимый шаг для её решения с помощью компьютера;
- составлять и записывать алгоритм с помощью определенных средств и методов описания; оценивать алгоритмы; применять на практике основные алгоритмические структуры
- линейную, условную и циклическую; разбивать сложные задачи на подзадачи;
- записывать алгоритмы обработки данных на изучаемом языке программирования Python, отлаживать и выполнять полученную программу в используемой среде разработки;
- использовать информационные и коммуникационные технологии для хранения, преобразования и передачи различных видов информации.

СОДЕРЖАНИЕ УЧЕБНОГО КУРСА «Программирование на языке Python: путь от первых проектов к реальным проектам (базовый уровень)»

В содержании учебного курса выделяются 4 модуля:

- «Алгоритмы и структуры данных»;
- «Практика программирования»;
- «Основы проектирования программного обеспечения»;
- «Разработка приложений, интегрированных в ИТ-инфраструктуру».

Алгоритмы и структуры данных

Оценка сложности алгоритмов на примере алгоритмов сортировки

Оценка сложности алгоритмов. Необходимость оценки сложности программ. Порядок роста. Константная, линейная, логарифмическая, квадратичная сложность. Наилучший, средний и наихудший случай. Оценка времени работы алгоритмов. Оценка алгоритма по памяти. Анализ алгоритмов с ветвлениями и циклами. Алгоритмы сортировки. Алгоритмы сортировки, основанные на сравнении: сортировка слиянием, быстрая сортировка. Оценка сложности алгоритмов сортировки. Оценка времени работы алгоритмов сортировки.

Структуры данных и динамическое программирование

Стек. Использование списка. Вычисление арифметических выражений с помощью стека. Проверка скобочных выражений. Очереди, деки.

Целочисленные алгоритмы. Использование связанных структур. Графы. «Жадные» алгоритмы. Алгоритм Дейкстры. Динамическое программирование

Работа со строками, файлами и графикой

Символьные строки. Операции со строками. Поиск в строках. Примеры обработки строк. Преобразование число-строка. Строки в процедурах и функциях. Рекурсивный перебор. Работа с файлами. Работа с текстовым файлом: чтение, запись, дозапись. Работа с бинарными файлами. Работа с графикой. Библиотеки для работы с графикой. Графические примитивы. Основные функции работы с графикой. Решение задач на моделирование.

Алгоритмы поиска

Последовательный поиск. Двоичный поиск в отсортированном массиве. Оценка сложности алгоритмов поиска.

Деревья поиска

Деревья. Двоичные деревья поиска. Обход дерева поиска. Поиск элемента. Поиск минимума и максимума. Вставка и удаление элементов. Сбалансированные деревья поиска. Обзор сбалансированных деревьев, АВЛ-дерево, Splay-дерево. Операции со сбалансированными деревьями поиска.

Хеширование

Хеш-таблицы с закрытой и открытой адресацией. Свойства хеш-таблицы. Хеширование в современных языках программирования.

Практика программирования

Решение и разбор олимпиадных задач. Типичные ошибки в программе и их исправление.

Рецензирование кода. Оформление программного кода в соответствии с установленными требованиями. Руководство по стилю. Техники рецензирования кода. Ветки в Git. Совместная работа. Проект на Github. Fork и Pull Request.

Проверка и отладка программного кода Разработка процедур проверки работоспособности и измерения характеристик программного обеспечения. Проверка работоспособности программного обеспечения

Объектно-ориентированное программирование. Классы. Объекты. Структуры. Наследование, полиморфизм, инкапсуляция. Шаблоны, интерфейсы.

Рефакторинг и оптимизация программного кода.

Основы проектирования программного обеспечения

Жизненный цикл программного обеспечения. Качество программного обеспечения. Анализ требований к программному обеспечению. Документирование программного обеспечения

Разработка технических спецификаций на программные компоненты и их взаимодействие. Обеспечение качества программных систем. Основные принципы проектирования программного обеспечения.

Разработка приложений, интегрированных в ИТ- инфраструктуру

Подключение и взаимодействие с системами управления базами данных. Протоколы передачи данных между компонентами клиент-серверной архитектуры. Разработка клиент-серверных приложений.

Особенности работы приложений под управлением различных операционных систем, сборка дистрибутивов приложений. Защищенное хранение данных в файлах с ограниченным доступом в различных операционных системах. Подготовка и защита учебного проекта

УЧЕБНЫЙ ПЛАН

Первый учебный год по 4 академических часа в неделю, всего 72 часа

№ п/п	Модуль	Наименование раздела	Всего	В том числе		Форма Аттестации (контроля)
				теория	практика	
1	Алгоритмы и структуры данных	Оценка сложности алгоритмов на примере алгоритмов сортировки	8	4	4	Тестирование
2	Алгоритмы и структуры данных	Элементарные структуры данных	8	4	4	Тестирование
3	Практика программирования	Работа со строками, файлами и графикой	8	4	4	Практическая работа
4	Алгоритмы и структуры данных	Алгоритмы поиска	8	4	4	Практическая работа
5	Алгоритмы и структуры данных	Деревья поиска	8	4	4	Практическая работа
6	Алгоритмы и структуры данных	Хеширование	6	4	2	Практическая работа
7	Практика программирования	Решение олимпиадных задач по алгоритмизации и программированию	10	4	6	Тестирование
8	Практика программирования	Совместная работа над проектом с использованием системы контроля версий	12	4	8	Практическая работа
Резерв учебного времени			4	0	4	
Общее количество часов			72	32	40	

Второй учебный год по 4 академических часа в неделю, всего 72 часа

№ п/п	Модуль	Наименование раздела	Всего	В том числе		Форма Аттестации (контроля)
				теория	практика	
1	Алгоритмы и структуры данных	Целочисленные алгоритмы. Использование связанных структур. Графы. «Жадные» алгоритмы. Алгоритм Дейкстры. Динамическое программирование	8	4	4	Тестирование
2	Практика программирования	Проверка и отладка программного кода Разработка процедур проверки работоспособности и измерения характеристик программного обеспечения. Проверка роботоспособности программного обеспечения	8	4	4	Тестирование
3	Основы проектирования программного обеспечения	Жизненный цикл программного обеспечения. Качество программного обеспечения. Анализ требований к программному обеспечению Документирование программного обеспечения	8	4	4	Практическая работа
4	Разработка приложений, интегрированных в ИТ-инфраструктуру	Подключение и взаимодействие с системами управления базами данных Протоколы передачи данных между компонентами клиент-серверной архитектуры. Разработка клиент-серверных приложений	8	4	4	Практическая работа
5	Практика программирования	Основы объектно-ориентированного программирования. Рефакторинг и оптимизация программного кода	10	4	6	Практическая работа
6	Основы проектирования программного обеспечения	Разработка технических спецификаций на программные компоненты и их взаимодействие Обеспечение качества программных систем Основные принципы	12	6	6	Тестирование

		проектирования программного обеспечения				
7	Разработка приложений, интегрированных в ИТ-инфраструктуру	Особенности работы приложений под управлением различных операционных систем, сборка дистрибутивов приложений. Защищенное хранение данных в файлах с ограниченным доступом в различных операционных системах. Подготовка и защита учебного проекта	14	6	8	Практическая работа
Резерв учебного времени			4	0	4	
Общее количество часов в 10 классе			72	32	40	

Разделы, относящиеся к одному модулю, могут быть реализованы в различных полугодиях. В том числе, возможно параллельное изучение материала нескольких модулей, если это обосновано логикой освоения материала.

УЧЕБНО - ТЕМАТИЧЕСКИЙ ПЛАН

Первый год обучения

Алгоритмы и структуры данных

Оценка сложности алгоритмов на примере алгоритмов сортировки

- Изучение основных понятий сложности алгоритмов: временная и пространственная сложность.
- Анализ сложности популярных алгоритмов сортировки (быстрая сортировка, сортировка слиянием, пузырьковая сортировка) и их сравнение.
- Практические задачи на реализацию алгоритмов сортировки и оценку их эффективности.

Элементарные структуры данных

- Изучение и реализация основных структур данных: массивы, списки, стеки, очереди.
- Примеры применения каждой структуры данных в программировании.
- Лабораторные работы на реализацию и использование элементарных структур данных в Python.

Алгоритмы поиска

- Обзор алгоритмов линейного и бинарного поиска.
- Реализация и анализ эффективности алгоритмов поиска.
- Практические задания на поиск данных в различных структурах.

Деревья поиска

- Введение в деревья и бинарные деревья поиска.
- Алгоритмы вставки, удаления и поиска в бинарном дереве поиска.
- Примеры реализации и применения деревьев поиска в задачах.

Хеширование

- Основы хеширования и применение хеш-таблиц.
- Реализация и использование хеш-функций в Python.
- Решение практических задач с использованием хеширования для повышения эффективности поиска.

Практика программирования

Работа со строками, файлами и графикой

- Основы работы со строками и методы строк в Python.
- Чтение и запись файлов, работа с файловой системой.
- Введение в библиотеки для работы с графикой и создание простых графических приложений.
- Решение олимпиадных задач по алгоритмизации и программированию
- Анализ типовых олимпиадных задач и стратегий их решения.

- Практика решения задач на алгоритмические конструкции в Python.
- Участие в онлайн-контестах для закрепления навыков программирования.

Совместная работа над проектом с использованием системы контроля версий

- Введение в системы контроля версий (Git) и их необходимость в разработке программного обеспечения.
- Работа с GitHub: создание репозитория, ветвление, слияние изменений.
- Реализация группового проекта с использованием инструментов контроля версий.

Второй год обучения

2. Алгоритмы и структуры данных

Оценка сложности алгоритмов на примере алгоритмов сортировки

- Введение в понятие сложности алгоритмов: время исполнения и занимаемая память.
- Асимптотические обозначения: О-большое, Ω (Омега) и Θ (Тета).
- Примеры расчета сложности алгоритмов сортировки: сортировка пузырьком, быстрая сортировка, сортировка слиянием.

Основные структуры данных: массивы, связные списки, стеки и очереди

- Описание и принцип работы каждой структуры данных.
- Примеры использования в программировании.
- Сложность основных операций для каждой структуры данных (доступ, вставка, удаление).

Деревья и графы: базовые понятия и примеры алгоритмов

- Введение в деревья: бинарные деревья поиска, балансировка деревьев.
- Обход деревьев: в глубину и в ширину.
- Введение в графы: представление графов (списки смежности, матрицы смежности).
- Примеры алгоритмов на графах: поиск в глубину и поиск в ширину.

Алгоритмы на графах: кратчайшие пути и алгоритмы поиска

- Алгоритм Дейкстры для поиска кратчайшего пути в взвешенном графе без отрицательных весов.
- Алгоритм Флойда-Уоршелла для поиска кратчайших путей между всеми парами вершин.
- Алгоритм поиска минимального остовного дерева (Прима или Краскала).

3. Алгоритмы и структуры данных

Жизненный цикл программного обеспечения

- Введение в жизненный цикл программного обеспечения (ПО): основные этапы и их характеристики.
- Модели жизненного цикла ПО: водопадная модель, итеративная модель, агил-

методологии (например, SCRUM).

- Значение каждого этапа в жизненном цикле и их влияние на конечный продукт.

Качество программного обеспечения

- Понятие качества ПО и факторы, влияющие на качество.
- Методы обеспечения качества ПО: тестирование, ревью кода, статический анализ кода.
- Стандарты качества ПО и сертификация: ISO/IEC 9126, ISO/IEC 25010.

Анализ требований к программному обеспечению

- Введение в анализ требований: зачем нужен анализ требований и какие проблемы решает.
- Методы сбора требований: интервью, опросы, анализ документации.
- Спецификация требований: написание функциональных и нефункциональных требований, применение стандартов (например, IEEE 830).

Документирование программного обеспечения

- Значение и цели документирования в проектах разработки ПО.
- Основные типы документации ПО: техническое задание, руководство пользователя, архитектурные описания.
- Инструменты и лучшие практики по созданию и поддержке документации.

3. Разработка приложений, интегрированных в ИТ- инфраструктуру

Подключение и взаимодействие с системами управления базами данных (СУБД)

- Обзор основных типов СУБД (реляционные, NoSQL) и их использование в современной разработке.
- Методы подключения к базам данных: использование драйверов, ORM (Object-Relational Mapping) библиотек.
- Практические примеры выполнения операций CRUD (создание, чтение, обновление, удаление данных) с использованием популярных СУБД (например, PostgreSQL, MongoDB).

Протоколы передачи данных между компонентами клиент-серверной архитектуры
Основы клиент-серверной архитектуры: роли клиента и сервера, преимущества и недостатки.

- Обзор ключевых протоколов передачи данных: HTTP, HTTPS, WebSocket.
- Примеры использования REST и GraphQL для обмена данными между клиентом и сервером.

Разработка клиент-серверных приложений

- Проектирование архитектуры клиент-серверного приложения: разделение ответственности, выбор технологий для клиента и сервера.

- Разработка API: проектирование, документирование (Swagger, OpenAPI) и версионирование.
- Безопасность в клиент-серверных приложениях: аутентификация, авторизация, защита от основных видов атак (SQL инъекции, XSS, CSRF).

Интеграция приложений в ИТ-инфраструктуру

- Введение в интеграцию: понятие и задачи интеграции приложений в корпоративной ИТ-инфраструктуре.
- Современные подходы к интеграции: микросервисы, API Gateway, использование брокеров сообщений (Kafka, RabbitMQ).
- Управление конфигурацией и развертывание приложений: контейнеризация (Docker), оркестрация контейнеров (Kubernetes), CI/CD пайплайны.

4. Практика программирования

Основы объектно-ориентированного программирования (ООП) (1 академический час)

- Введение в ООП: основные концепции и принципы (инкапсуляция, наследование, полиморфизм).
- Классы и объекты: определение, создание и использование в программировании.

Принципы SOLID в объектно-ориентированном программировании

- Обзор и значение принципов SOLID для создания гибкого и поддерживаемого кода.
- Подробное разбор каждого принципа с примерами: SRP (Принцип единственной ответственности), OCP (Принцип открытости/закрытости), LSP (Принцип подстановки Барбары Лисков), ISP (Принцип разделения интерфейса), DIP (Принцип инверсии зависимостей).
- Практические задания на применение принципов SOLID в коде.

Рефакторинг кода: методы и техники

- Введение в рефакторинг: цели, задачи и принципы рефакторинга кода.
- Распространенные паттерны и приемы рефакторинга: выделение метода, переименование переменных, удаление дублирующегося кода.
- Инструменты для рефакторинга и анализа кода: IDE фичи, статические анализаторы кода (например, SonarQube).

Оптимизация программного кода

- Понятие и необходимость оптимизации кода: производительность и эффективность использования ресурсов.
- Анализ и профилирование приложений: инструменты и методы для выявления узких мест.
- Практические примеры оптимизации: оптимизация циклов, использование

коллекций, минимизация обращений к базе данных.

5. Основы проектирования программного обеспечения

Особенности работы приложений под управлением различных операционных систем, сборка дистрибутивов приложений

- Мультиплатформенная разработка: стратегии и инструменты (например, Docker, VirtualBox).
- Сборка и дистрибьюция приложений: системы сборки (Maven, Gradle для Java, CMake для C/C++), контейнеризация.
- Тестирование приложений в различных операционных системах.

Защищенное хранение данных в файлах с ограниченным доступом в различных операционных системах

- Основы криптографии и шифрования данных.
- Методы реализации защищенного хранения данных: файловые системы с шифрованием, API операционных систем для работы с защищенными данными.
- Практические аспекты обеспечения безопасности данных в приложениях.

6. Разработка приложений, интегрированных в ИТ- инфраструктуру

Подготовка и защита учебного проекта

- Планирование и подготовка учебного проекта: от выбора темы до разработки плана реализации.
- Разработка презентации проекта: структура, ключевые моменты, демонстрация работы приложения.
- Советы и рекомендации по успешной защите проекта перед аудиторией.

КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК

Начало занятий	По мере формирования группы обучающихся
Окончание обучения	24 месяца при обучении 2 раза в неделю по одному академическому часу
Срок обучения	24 месяца при обучении 2 раза в неделю по одному академическому часу
Формы учебной работы	Аудиторные занятия (тренинги)
Количество занятий в неделю	2 занятия
Начало занятий	Согласно расписанию
Режим занятий	4 академических часа
Форма обучения	Очная

ОБЩИЕ ТРЕБОВАНИЯ К ПЕДАГОГУ

1. Образование и квалификация:
   - Высшее профессиональное образование в области информационных технологий, компьютерных наук, прикладной математики или смежных направлений.
   - Дополнительная квалификация или сертификация, подтверждающая знания в области программирования на Python.
2. Опыт работы:
   - Опыт программирования на Python не менее 2 лет.
   - Опыт преподавания или проведения тренингов по программированию для школьников или студентов будет являться преимуществом.
   - Реальный опыт разработки программного обеспечения или участие в IT-проектах.
3. Педагогические навыки:
   - Способность объяснять сложные понятия простым и доступным языком.
   - Умение мотивировать учащихся, поддерживать интерес к изучаемому предмету.
   - Навыки организации интерактивного и практико-ориентированного обучения.
4. Личностные качества:
   - Ответственность, терпимость, умение находить подход к разным категориям учащихся.
   - Готовность к постоянному профессиональному и личностному развитию.
   - Креативность в подходах к обучению и использовании различных образовательных технологий.
5. Технические навыки:
   - Знание современных образовательных технологий и умение их эффективно применять в учебном процессе.
   - Владение инструментами для создания и распространения учебных материалов, тестирования и мониторинга успеваемости учащихся.
   - Умение работать с различными операционными системами, программным обеспечением для программирования и разработки.
6. Языковые навыки:
   - Владение русским языком на уровне носителя для эффективной коммуникации с учащимися и родителями.
   - Знание английского языка на уровне, позволяющем свободно читать техническую документацию и использовать международные образовательные ресурсы.

Педагог должен быть способен адаптировать учебный процесс под индивидуальные особенности и потребности учащихся, используя разнообразные формы и методы обучения.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ И ОБОРУДОВАНИЯ

ИСПОЛЬЗУЕМЫЙ УМК, включенный в Федеральный перечень учебников (приказ №858 от 21.09.2022 г.):

УМК «Информатика. 9 класс. Учебник. ФГОС» Поляков К.Ю., Еремин Е.А.

УМК «Информатика. 10 – 11 классы. Углубленный уровень (в 2 частях)» Поляков К.Ю., Еремин Е.А.

Основные дополнительно используемые учебные пособия:

«Программирование. Python. Часть 1» К.Ю. Поляков

«Программирование. Python. Часть 2» К.Ю. Поляков

«Программирование. Python. Часть 3» К.Ю. Поляков

«Программирование. Python. Часть 4» К.Ю. Поляков

ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМОГО ОБОРУДОВАНИЯ

1. Персональный компьютер с периферией/ноутбук (лицензионное программное обеспечение, образовательный контент, система защиты от вредоносной информации) – рабочее место ученика.
2. Персональный компьютер с периферией/ноутбук (лицензионное программное обеспечение, образовательный контент, система защиты от вредоносной информации) – рабочее место учителя.
3. Тележка-хранилище ноутбуков/планшетов с системой подзарядки в комплекте с ноутбуками/планшетами (лицензионное программное обеспечение, образовательный контент, система защиты от вредоносной информации, программное обеспечение с возможностью подготовки к ГИА, программное обеспечение для цифровых лабораторий).
4. Интерактивная доска.
5. Проектор.
6. Экран для проетора
7. Принтер (струйный / лазерный).
8. Сканер.
9. Сервер.
10. Комплект сетевого оборудования.
11. Комплект оборудования для подключения к сети Интернет.
12. Цифровой фотоаппарат.
13. Цифровая видеокамера.
14. Устройство для чтения информации с карты памяти (картридер).
15. Web-камера.
16. Устройства ввода/вывода звуковой информации – микрофон, колонки и наушники.
17. Внешний накопитель информации.
18. Мобильное устройство для хранения информации (флеш-память).