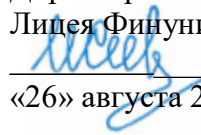


Федеральное государственное образовательное бюджетное
учреждение высшего образования
«Финансовый университет при Правительстве Российской Федерации»

УТВЕРЖДАЮ

Директор

Лицея Финуниверситета

 И.В. Сивцова

«26» августа 2026 г.

РАБОЧАЯ ПРОГРАММА КУРСА ВНЕУРОЧНОЙ ДЕЯТЕЛЬНОСТИ
«ПРОГРАММИРОВАНИЕ»
Среднее общее образование

Москва – 2026 г.

Рабочая программа согласована
на заседании педагогического совета
Протокол № 1 от «26» августа 2026 г.

Рабочая программа курса внеурочной деятельности

«Программирование»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Рабочая программа курса внеурочной деятельности «Программирование» рассчитана на обучающихся 10–11 классов Лицея Финансового университета.

В современном мире, в котором цифровые технологии пронизывают все сферы жизни — от финансов и экономики до науки и образования, — умение программировать становится не просто полезным навыком, а фундаментальной составляющей успешной профессиональной деятельности. Особенно велика его роль при обучении в предуниверситетской Финуниверситета, где формируются будущие специалисты в области анализа данных, финансовых технологий и информационной безопасности.

В процессе освоения курса обучающиеся не только изучают синтаксис и конструкции языка Python, но и овладевают методами решения алгоритмических задач, знакомятся с принципами объектно-ориентированного программирования и работой с данными. Изучение программирования в выпускных классах служит систематизации полученных ранее знаний, формирует базу для дальнейшего изучения ИТ-дисциплин в вузе и помогает в осознанном выборе карьеры в рамках экономики данных.

Программа курса ориентирует педагога на последовательное формирование у обучающихся прочных навыков алгоритмического мышления и написания кода. Вся программа поделена на содержательные модули — от основ программирования до продвинутых тем и проектной деятельности — что позволяет логично выстраивать процесс обучения с нарастающей сложностью. При освоении модулей особое внимание уделяется практической отработке каждой темы, разбору типичных ошибок и накоплению опыта решения задач различного уровня (от учебных до близких к олимпиадным).

При решении задач по программированию акцент делается на понимании алгоритма, выборе оптимальных структур данных, чистоте и читаемости кода, а также на навыках отладки и документирования.

При повторении обобщаются и систематизируются как теоретические основы языка, так и практические приемы кодинга, принимаются во внимание цели подготовки к участию в олимпиадах и дальнейшему обучению в вузе.

Цель программы

- Сформировать у обучающихся устойчивые навыки алгоритмического мышления и программирования на языке Python для решения широкого круга задач — от учебных до прикладных.
- Обеспечить дополнительную инструментальную подготовку обучающихся для участия в олимпиадах по информатике и программированию (эта часть программы предусматривает решение задач как базового, так и повышенного и высокого уровня сложности).
- Развить содержание курса информатики для углубленного изучения современных ИТ-технологий, необходимых в финансовой сфере (анализ данных, автоматизация, защита информации).

В соответствии с планом внеурочной деятельности Лицея на освоение курса внеурочной деятельности «Программирование» отводится 136 часов. Рабочая программа предусматривает обучение в объеме 2 часа в неделю в течение 2 учебных лет (10–11 классы). Распределение часов для изучения различных разделов программы не является жестко детерминированным. Оно может варьироваться в зависимости от подготовленности и запросов обучающихся, а также от глубины погружения в ту или иную тему.

Используемые учебники и пособия

1. Поляков К.Ю. Программирование. Python. C++. Учебное пособие. В 2 ч. — М.: Просвещение, 2025.
2. Чернышев С.А. Алгоритмы и структуры данных на Python: учебное пособие. — М.: КноРус, 2024. — 326 с. — ISBN 978-5-406-11683-8.
3. Горохова Р.И. Практикум по программированию на языке Python: учебное пособие для студентов вузов. — М.: Финансовый университет, 2024.
4. Зингаро Д. Python без проблем. Решаем реальные задачи и пишем полезный код. — СПб.: Питер, 2025. — 352 с. — ISBN 978-5-4461-1920-2.
5. Любанович Б. Простой Python. Современный стиль программирования. — 2-е изд. — СПб.: Питер, 2024. — 480 с.

Литература и Интернет-ресурсы

1. <https://stepik.org/course/58852> — онлайн-курс «Поколение Python: курс для начинающих» на платформе Stepik (бесплатный, структурированный, с автоматической проверкой задач).
2. <https://informatics.msk.ru> — образовательная платформа с теоретическими материалами и задачами по программированию для школьников.

3. <https://codeforces.com> — международная платформа для соревнований по программированию, архив задач и тренировок (в том числе для подготовки к олимпиадам).

РЕЗУЛЬТАТЫ ОСВОЕНИЯ КУРСА ВНЕУРОЧНОЙ ДЕЯТЕЛЬНОСТИ «ПРОГРАММИРОВАНИЕ»

Личностные результаты

- в ценностно-ориентационной сфере – чувство гордости за достижения российских учёных и инженеров в области информатики и программирования, гуманизм, положительное отношение к труду, целеустремленность;
- в трудовой сфере — готовность к осознанному выбору дальнейшей образовательной траектории;
- в познавательной (когнитивной, интеллектуальной) сфере — умение управлять своей познавательной деятельностью.

Метапредметные результаты

- овладение умениями проводить анализ поставленной задачи, подбирать и реализовывать методы решения, анализировать результаты;
- развитие познавательных интересов, интеллектуальных и творческих способностей в процессе решения задач по программированию и самостоятельного приобретения новых знаний, расширение границ применимости знаний в области программирования;
- воспитание духа сотрудничества в процессе совместного выполнения задач, уважения к творцам науки и техники, обеспечивающим ведущую роль информатики и программирования в современном мире;
- использование приобретенных знаний и умений для решения практических, жизненных задач, обеспечения информационной безопасности и обеспечения безопасности жизнедеятельности человека и общества.

Предметные результаты

Обучающиеся должны уметь:

- анализировать алгоритмическую проблему;
- проговаривать вслух алгоритм решения;
- анализировать полученный результат выполнения программы;
- классифицировать предложенную задачу по типу алгоритма и структурам данных;
- составлять простейшие программы;
- последовательно выполнять и проговаривать этапы разработки программы средней сложности;

- выбирать рациональный способ решения задачи (оптимальные структуры данных, алгоритмы);
- решать комбинированные задачи, требующие интеграции нескольких тем;
- владеть различными методами программирования: итеративным, рекурсивным, объектно-ориентированным.

СОДЕРЖАНИЕ КУРСА ВНЕУРОЧНОЙ ДЕЯТЕЛЬНОСТИ «ПРОГРАММИРОВАНИЕ» С УКАЗАНИЕМ ФОРМ ОРГАНИЗАЦИИ И ВИДОВ ДЕЯТЕЛЬНОСТИ

1. Основы программирования на Python

Переменные и типы данных (числа, строки, булевы значения). Ввод и вывод данных (функции `input()`, `print()`). Условный оператор `if-elif-else`. Логические операторы и операции сравнения. Циклы `for` и `while`, операторы `break`, `continue`. Функции: определение, параметры, возврат значений, области видимости. Строки: индексация, срезы, основные методы (`split()`, `join()`, `upper()`, `lower()`). Работа со стандартной библиотекой: модули `random`, `math`, `datetime`. Обработка ошибок (конструкция `try-except`).

2. Алгоритмы и структуры данных

Списки: создание, индексация, методы (`append()`, `insert()`, `remove()`, `pop()`, `sort()`). Срезы списков. Генераторы списков (`list comprehension`). Кортежи и множества (операции над множествами). Словари: создание, методы `keys()`, `values()`, `items()`, обход. Рекурсия: базовый случай, рекурсивный шаг, примеры (факториал, числа Фибоначчи). Алгоритмы поиска: линейный поиск, бинарный поиск (итеративная и рекурсивная версии). Алгоритмы сортировки: сортировка пузырьком, сортировка выбором, быстрая сортировка (`quicksort`). Оценка сложности алгоритмов (O-большое) — базовое понятие.

3. Объектно-ориентированное программирование

Понятие класса и объекта. Конструктор `__init__`. Атрибуты экземпляра и методы класса. Наследование: создание дочерних классов, переопределение методов. Инкапсуляция: приватные атрибуты и методы, использование `property`. Полиморфизм: единый интерфейс для разных классов. «Магические» методы (`__str__`, `__repr__`, `__eq__`, `__add__`). Статические методы и методы класса (`@staticmethod`, `@classmethod`). Примеры: класс «Банковский счёт», класс «Геометрическая фигура» (с наследованием — круг, квадрат, треугольник), класс «Студент».

4. Работа с файлами и внешними данными

Чтение и запись текстовых файлов (режимы r, w, a). Работа с форматом CSV (модуль csv). Формат JSON: сериализация и десериализация с помощью модуля json. Отправка HTTP-запросов (библиотека requests), получение данных с публичных API (курсы валют, погода). Обработка и сохранение полученных данных. Визуализация данных с помощью matplotlib (построение графиков, гистограмм, диаграмм рассеяния). Основы работы с pandas: загрузка DataFrame, первичный анализ данных (описательные статистики, фильтрация).

5. Системы контроля версий и командная разработка (ознакомительно)

Основы Git: инициализация репозитория (git init), добавление и фиксация изменений (add, commit), просмотр истории (log). Ветвление и слияние (branch, merge). Разрешение конфликтов. Работа с удалённым репозиторием (GitHub): клонирование (clone), отправка изменений (push), получение обновлений (pull). Понятие pull request и код-ревью. Использование .gitignore.

Итоговая проектная деятельность

Выбор темы проекта (индивидуально или в группе 2–3 человека). Этапы разработки: постановка задачи, проектирование архитектуры, написание кода, тестирование, документирование. Создание репозитория на GitHub. Подготовка презентации и публичная защита проекта (демонстрация работы, ответы на вопросы). Взаимооценка проектов по рубрике.

Формы организации и виды деятельности

- проведение мини-лекции с разбором теоретического материала;
- проведение практического занятия (решение задач в среде разработки, написание кода);
- работа в парах (парное программирование);
- выполнение домашних заданий с автоматизированной проверкой (Stepik, Replit);
- разбор и отладка кода (code review);
- игровые форматы: «алгоритмическая эстафета», «найди ошибку в коде»;
- выполнение мини-проектов (класс, модуль, утилита);
- работа с системами контроля версий (Git);

- защита итогового проекта (презентация + демонстрация).

ТЕМАТИЧЕСКОЕ ПЛАНИРОВАНИЕ

№ п/п	Тема	Форма деятельности	Количество часов на изучение	Уровень
Модуль 1. Основы программирования на Python (10 класс)				
1.	Входная диагностика: написание простейшей программы	практическое занятие	2	Б
2.	Переменные, типы данных, ввод/вывод (input, print)	лекция + практикум	2	Б
3.	Условный оператор if-elif-else, логические операторы	практическое занятие	2	Б
4.	Цикл for, функция range(), итерация по коллекциям	практическое занятие	2	Б
5.	Цикл while, операторы break, continue	практическое занятие	2	Б
6.	Строки: индексация, срезы, методы (split, join, upper, lower)	практическое занятие	2	Б
7.	Функции: определение, параметры, возврат значений	лекция + практикум	2	Б
8.	Области видимости переменных (global, local)	практическое занятие	2	Б
9.	Модули random, math, datetime	практическое занятие	2	Б
10.	Обработка ошибок (try-except)	практическое занятие	2	Б
11.	Базовые алгоритмы: линейный поиск, поиск максимума/минимума	практическое занятие	2	Б
12.	Сортировка пузырьком и сортировка выбором	лекция + практикум	2	Б
13.	Бинарный поиск (итеративная версия)	практическое занятие	2	Б
14.	Рубежный тест (модуль 1)	тестирование	2	Б
Модуль 2. Структуры данных и рекурсия (10 класс)				
15.	Списки: создание, индексация, методы (append, insert, remove)	лекция + практикум	2	П
16.	Срезы списков, копирование списков	практическое занятие	2	П
17.	Генераторы списков (list	практическое занятие	2	П

	comprehension)			
18.	Кортежи и множества (set, frozenset)	практическое занятие	2	П
19.	Словари: создание, методы keys(), values(), items()	практическое занятие	2	П
20.	Генераторы словарей	практическое занятие	2	П
21.	Рекурсия: базовый случай, рекурсивный шаг	лекция + практикум	2	П
22.	Рекурсивные алгоритмы: факториал, числа Фибоначчи	практическое занятие	2	П
23.	Рекурсивный бинарный поиск	практическое занятие	2	П
24.	Быстрая сортировка (quicksort)	лекция + практикум	2	П
25.	Понятие сложности алгоритмов (O-большое) – вводное	лекция + обсуждение	2	П
26.	Рубежный тест (модуль 2)	тестирование	2	П
Модуль 3. Объектно-ориентированное программирование (10–11 классы)				
27.	Понятие класса и объекта. Конструктор __init__	лекция + практикум	2	П
28.	Атрибуты экземпляра и методы класса	практическое занятие	2	П
29.	Наследование: создание дочерних классов, переопределение методов	практическое занятие	2	П
30.	Инкапсуляция: приватные атрибуты, property	практическое занятие	2	П
31.	Полиморфизм: единый интерфейс для разных классов	практическое занятие	2	П
32.	Магические методы (__str__, __repr__, __eq__)	практическое занятие	2	П
33.	Магические методы (__add__, __len__)	практическое занятие	2	П
34.	Статические методы и методы класса (@staticmethod, @classmethod)	практическое занятие	2	П
35.	Пример: класс «Банковский счёт»	практическое занятие (мини-проект)	2	П
36.	Пример: класс	работа в парах	2	П

	«Геометрическая фигура» (наследование)			
37.	Классы-исключения: создание собственных исключений	лекция + практикум	2	П
38.	Композиция и агрегация классов	практическое занятие	2	П
39.	Рубежный тест (ООП)	тестирование	2	П
Модуль 4. Работа с файлами и внешними библиотеками (11 класс)				
40.	Чтение и запись текстовых файлов (режимы r, w, a)	лекция + практикум	2	В
41.	Работа с форматом CSV (модуль csv)	практическое занятие	2	В
42.	Формат JSON: сериализация и десериализация (json)	практическое занятие	2	В
43.	Отправка HTTP-запросов (библиотека requests)	лекция + практикум	2	В
44.	Получение данных с публичных API (курсы валют, погода)	практическое занятие	2	В
45.	Обработка и сохранение полученных данных из API	практическое занятие	2	В
46.	Визуализация данных с matplotlib: построение графиков	практическое занятие	2	В
47.	Визуализация: гистограммы, диаграммы рассеяния	практическое занятие	2	В
48.	Основы pandas: загрузка DataFrame, первичный анализ	лекция + практикум	2	В
49.	pandas: описательные статистики, фильтрация, группировка	практическое занятие	2	В
50.	Проектная задача: анализ данных с визуализацией	работа в парах	2	В
Модуль 5. Системы контроля версий и командная разработка (11 класс)				
51.	Введение в Git: установка, инициализация репозитория (git init)	лекция + демонстрация	2	В
52.	Основные команды: add, commit, log, status	практическое занятие	2	В
53.	Ветвление: создание и переключение веток	практическое занятие	2	В

	(branch, checkout)			
54.	Слияние веток (merge), разрешение конфликтов	практическое занятие	2	В
55.	Работа с удалённым репозиторием GitHub: clone, push, pull	практическое занятие	2	В
56.	Понятие pull request и код-ревью	лекция + обсуждение	2	В
57.	Файл .gitignore, работа с чужими репозиториями (fork)	практическое занятие	2	В
Модуль 6. Итоговая проектная деятельность (11 класс)				
58.	Выбор темы проекта, постановка задачи, обоснование	консультация, индивидуальная работа	2	В
59.	Проектирование архитектуры, выбор библиотек	работа в группах	2	В
60.	Разработка проекта: написание кода, создание репозитория	практическое занятие	4	В
61.	Продолжение разработки, промежуточное ревью	практическое занятие	4	В
62.	Тестирование, отладка, документирование кода	практическое занятие	4	В
63.	Подготовка презентации и демонстрации	практическое занятие	2	В
64.	Защита проектов (публичная демонстрация, вопросы)	защита проекта	2	Ф
65.	Итоговая рефлексия курса, самооценка	круглый стол	2	Ф
Резервные часы (повторение, консультации, отработка пропущенных тем)				
66–68	Резерв – повторение ключевых тем, консультации по проектам	различные формы	6	Ф
	Итого		136 часов	

Обозначения уровня сложности:

Б — базовый

П — продвинутый

В — высокий

Ф — повторение / финализация

Примечание: Резервные часы (6 часов) могут быть использованы для углублённого разбора сложных тем, дополнительных практических занятий

или помощи отстающим. Распределение часов по темам может незначительно варьироваться в зависимости от уровня подготовленности группы.

Методические рекомендации

Особенность данного курса состоит в том, что прежде всего обучающиеся осваивают базовый синтаксис и конструкции языка Python (переменные, условия, циклы, функции), а затем последовательно переходят к более сложным темам: структурам данных, рекурсии, объектно-ориентированному программированию, работе с файлами и API, а также к системам контроля версий. Курс построен по принципу «от простого к сложному»: каждая тема опирается на ранее изученный материал. Большое внимание уделяется практическому программированию: решению задач, отладке, написанию небольших проектов. Важно формировать у обучающихся культуру написания читаемого, документированного кода и использования систем контроля версий.

I. Модуль 1. Основы программирования на Python (10 класс)

1. Входная диагностика: написание простейшей программы

Обучающиеся без предварительной подготовки пишут программу, которая запрашивает имя и выводит приветствие.

Необходимо повторить: отсутствие специальных знаний – диагностика исходного уровня.

2. Переменные, типы данных, ввод/вывод (input, print)

Рассматриваются основные типы данных: целые числа, числа с плавающей точкой, строки, булевы значения. Осваиваются функции input() и print(), форматирование вывода (f-строки).

Необходимо повторить: понятие переменной, отличие типа данных, правила именования.

3. Условный оператор if-elif-else, логические операторы

Изучается синтаксис ветвления, операторы сравнения (==, !=, <, >, <=, >=), логические операторы (and, or, not). Решаются задачи с несколькими условиями.

Необходимо повторить: логические выражения, вложенные условия.

4. Цикл for, функция range(), итерация по коллекциям

Осваивается цикл с фиксированным числом повторений, функция range() с одним, двумя и тремя аргументами. Итерация по строкам и спискам.

Необходимо повторить: понятие итерации, отличие цикла от ветвления.

5. Цикл while, операторы break, continue

Изучается цикл с предусловием, его применение в задачах, где число повторений неизвестно заранее. Операторы досрочного выхода и пропуска итерации.

Необходимо повторить: условия завершения цикла, предотвращение бесконечных циклов.

6. Строки: индексация, срезы, методы (split, join, upper, lower)

Рассматриваются строки как последовательности символов, доступ по индексу, извлечение подстрок (срезы). Базовые методы для преобразования и разбора строк.

Необходимо повторить: индексация с нуля, отрицательные индексы, неизменяемость строк.

7. Функции: определение, параметры, возврат значений

Изучается синтаксис объявления функций, передача аргументов, оператор return. Понятие локальных переменных.

Необходимо повторить: различие между определением и вызовом функции, преимущества модульности.

8. Области видимости переменных (global, local)

Рассматриваются зоны программного кода, в которых переменная доступна. Ключевое слово global для изменения глобальной переменной внутри функции.

Необходимо повторить: иерархия областей видимости (локальная, enclosing, глобальная, встроенная).

9. Модули random, math, datetime

Знакомство со стандартной библиотекой. Генерация случайных чисел, математические функции (корень, степени, тригонометрия), работа с датой и временем.

Необходимо повторить: импорт модуля (import, from ... import ...).

10. Обработка ошибок (try-except)

Рассматривается перехват исключений, чтобы программа не падала при некорректном вводе. Блоки try, except, else, finally.

Необходимо повторить: типы исключений (ValueError, ZeroDivisionError, TypeError).

11. Базовые алгоритмы: линейный поиск, поиск максимума/минимума

Реализация на Python линейного поиска элемента в списке, поиска максимального и минимального значения.

Необходимо повторить: итерация по списку, сравнение элементов.

12. Сортировка пузырьком и сортировка выбором

Изучаются простые алгоритмы сортировки. Анализируется количество сравнений и обменов. Реализация на Python.

Необходимо повторить: вложенные циклы, обмен значений переменных.

13. Бинарный поиск (итеративная версия)

Алгоритм поиска элемента в отсортированном массиве за $O(\log n)$. Реализация с помощью цикла while.

Необходимо повторить: условие отсортированности, деление пополам.

14. Рубежный тест (модуль 1)

Проверка усвоения тем модуля 1. Тест включает теоретические вопросы и небольшие практические задачи.

II. Модуль 2. Структуры данных и рекурсия (10 класс)

15. Списки: создание, индексация, методы (append, insert, remove)

Изучается основной тип данных для хранения последовательностей. Методы добавления, вставки, удаления элементов.

Необходимо повторить: изменяемость списков, отличие от строк.

16. Срезы списков, копирование списков

Извлечение подсписков, создание поверхностных копий (через срез и метод `copy()`).

Необходимо повторить: различие между присваиванием и копированием.

17. Генераторы списков (list comprehension)

Компактная запись создания списков на основе другого итерируемого объекта. Примеры: квадраты чисел, фильтрация.

Необходимо повторить: условные выражения внутри генератора.

18. Кортежи и множества (set, frozenset)

Неизменяемые последовательности (кортежи). Множества – неупорядоченные коллекции уникальных элементов, операции объединения, пересечения, разности.

Необходимо повторить: отличие от списков, хешируемость элементов.

19. Словари: создание, методы keys(), values(), items()

Пара «ключ-значение». Создание, доступ, добавление, удаление пар. Итерация по ключам, значениям, парам.

Необходимо повторить: требования к ключам (неизменяемый тип), быстрота поиска.

20. Генераторы словарей

Аналогично генераторам списков, но для создания словарей.

Необходимо повторить: синтаксис {key_expr: value_expr for item in iterable}.

21. Рекурсия: базовый случай, рекурсивный шаг

Понятие рекурсивной функции: вызов функцией самой себя. Базовый случай – условие остановки.

Необходимо повторить: стек вызовов, опасность бесконечной рекурсии.

22. Рекурсивные алгоритмы: факториал, числа Фибоначчи

Классические примеры рекурсивных функций. Сравнение с итеративным решением.

Необходимо повторить: эффективность (экспоненциальный рост для Фибоначчи).

23. Рекурсивный бинарный поиск

Реализация бинарного поиска через рекурсию: функция принимает границы подмассива.

Необходимо повторить: условие завершения, когда границы сходятся.

24. Быстрая сортировка (quicksort)

Алгоритм «разделяй и властвуй». Выбор опорного элемента, разделение на меньшие и большие, рекурсивная сортировка частей.

Необходимо повторить: рекурсия, слияние результатов.

25. Понятие сложности алгоритмов (О-большое) – вводное

Базовое представление о времени выполнения алгоритмов (константное, линейное, квадратичное, логарифмическое). Примеры.

Необходимо повторить: сравнение скорости сортировок и поисков.

26. Рубежный тест (модуль 2)

Проверка усвоения структур данных и рекурсии.

III. Модуль 3. Объектно-ориентированное программирование (10–11 классы)

27. Понятие класса и объекта. Конструктор __init__

Класс как шаблон, объект как экземпляр. Метод __init__ для инициализации атрибутов.

Необходимо повторить: отличие класса от функции, ключевое слово self.

28. Атрибуты экземпляра и методы класса

Создание и использование атрибутов, методы, которые работают с ними.

Необходимо повторить: синтаксис вызова метода `obj.method()`.

29. Наследование: создание дочерних классов, переопределение методов

Один класс наследует атрибуты и методы другого. Переопределение методов в дочернем классе.

Необходимо повторить: функция `super()` для вызова родительского метода.

30. Инкапсуляция: приватные атрибуты, `property`

Соккрытие внутренних данных. Именованное с двумя подчёркиваниями (`__attr`). Использование `@property` для контролируемого доступа.

Необходимо повторить: принцип «не изменяй внутренности чужого объекта напрямую».

31. Полиморфизм: единый интерфейс для разных классов

Разные классы могут иметь методы с одинаковыми именами, но разной реализацией.

Необходимо повторить: вызов метода на объекте без знания его конкретного типа.

32. Магические методы (`__str__`, `__repr__`, `__eq__`)

Методы, начинающиеся и заканчивающиеся на два подчёркивания. `__str__` – строковое представление для пользователя, `__repr__` – для разработчика, `__eq__` – сравнение.

Необходимо повторить: автоматический вызов этих методов (например, `print(obj)` вызывает `__str__`).

33. Магические методы (`__add__`, `__len__`)

Перегрузка операторов: `__add__` для `+`, `__len__` для `len()`.

Необходимо повторить: создание классов, имитирующих поведение встроенных типов.

34. Статические методы и методы класса (`@staticmethod`, `@classmethod`)

Методы, не требующие экземпляра (`@staticmethod`) или получающие класс вместо экземпляра (`@classmethod`).

Необходимо повторить: различие между методами экземпляра, класса и статическими.

35. Пример: класс «Банковский счёт»

Мини-проект: атрибуты (владелец, баланс), методы (пополнение, снятие, перевод), проверка корректности.

Необходимо повторить: все изученные принципы ООП в одном примере.

36. Пример: класс «Геометрическая фигура» (наследование)

Базовый класс «Фигура» с методами `area()` и `perimeter()`. Дочерние классы «Квадрат», «Круг», «Прямоугольник».

Необходимо повторить: переопределение методов, полиморфизм.

37. Классы-исключения: создание собственных исключений

Определение своего исключения наследованием от `Exception`. Выброс (`raise`) и перехват.

Необходимо повторить: стандартные исключения, конструкция `try-except`.

38. Композиция и агрегация классов

Отношение «часть-целое»: один класс содержит экземпляры другого. Отличие от наследования.

Необходимо повторить: передача объектов в конструктор, хранение в списке.

39. Рубежный тест (ООП)

Проверка знаний по всем темам ООП.

IV. Модуль 4. Работа с файлами и внешними библиотеками (11 класс)

40. Чтение и запись текстовых файлов (режимы `r`, `w`, `a`)

Открытие файла, методы `read()`, `readlines()`, `write()`. Использование менеджера контекста `with`.

Необходимо повторить: кодировки (UTF-8), закрытие файла.

41. Работа с форматом CSV (модуль `csv`)

Чтение и запись табличных данных. Итератор `csv.reader`, `csv.writer`. Словари `csv.DictReader`.

Необходимо повторить: структура CSV (разделители, заголовки).

42. Формат JSON: сериализация и десериализация (`json`)

Преобразование объектов Python в строку JSON (`json.dumps`) и обратно (`json.loads`). Работа с файлами (`dump`, `load`).

Необходимо повторить: типы данных в JSON (числа, строки, массивы, объекты).

43. Отправка HTTP-запросов (библиотека `requests`)

GET-запросы, получение ответа, проверка статус-кода.

Необходимо повторить: понятие URL, параметры запроса.

44. Получение данных с публичных API (курсы валют, погода)

Регистрация (при необходимости) и получение API-ключа. Разбор JSON-ответа.

Необходимо повторить: документация API, обработка ошибок сети.

45. Обработка и сохранение полученных данных из API

Извлечение нужных полей, сохранение в CSV или JSON.

Необходимо повторить: преобразование форматов.

46. Визуализация данных с matplotlib: построение графиков

Основы библиотеки: создание фигуры, оси, построение линейного графика (plot), настройка подписей.

Необходимо повторить: списки чисел, импорт модулей.

47. Визуализация: гистограммы, диаграммы рассеяния

Построение гистограмм (hist) и scatter-диаграмм (scatter).

Необходимо повторить: интерпретация графиков.

48. Основы pandas: загрузка DataFrame, первичный анализ

Структура DataFrame, загрузка из CSV, просмотр (head, tail, info).

Необходимо повторить: индексация строк и столбцов.

49. pandas: описательные статистики, фильтрация, группировка

Методы describe(), mean(), median(). Фильтрация строк по условию. Группировка groupby и агрегация.

Необходимо повторить: работа с пропусками (dropna, fillna).

50. Проектная задача: анализ данных с визуализацией

Комплексное задание: взять открытый датасет, очистить, построить графики, сделать выводы.

Необходимо повторить: все навыки модуля 4.

V. Модуль 5. Системы контроля версий и командная разработка (11 класс)

51. Введение в Git: установка, инициализация репозитория (git init)

Установка Git, базовая настройка (имя, email). Создание локального репозитория.

Необходимо повторить: понятие репозитория, отличие от обычной папки.

52. Основные команды: add, commit, log, status

Добавление файлов в индекс, фиксация изменений, просмотр истории и статуса.

Необходимо повторить: осмысленные сообщения коммитов.

53. Ветвление: создание и переключение веток (branch, checkout)

Создание параллельных веток для разработки новых функций. Переключение между ветками.

Необходимо повторить: зачем нужны ветки (изоляция изменений).

54. Слияние веток (merge), разрешение конфликтов

Объединение изменений из одной ветки в другую. Разрешение конфликтов вручную.

Необходимо повторить: ситуации конфликта (изменение одних и тех же строк).

55. Работа с удалённым репозиторием GitHub: clone, push, pull

Регистрация на GitHub, клонирование репозитория, отправка локальных изменений на сервер, получение обновлений.

Необходимо повторить: понятие origin, SSH/HTTPS.

56. Понятие pull request и код-ревью

Создание pull request на GitHub, обсуждение изменений, вливание после одобрения.

Необходимо повторить: процесс командной разработки.

57. Файл .gitignore, работа с чужими репозиториями (fork)

Исключение временных файлов, папок с секретами. Форк (копирование) репозитория, предложение изменений через pull request.

Необходимо повторить: открытые лицензии, этика открытого кода.

VI. Модуль 6. Итоговая проектная деятельность (11 класс)

58. Выбор темы проекта, постановка задачи, обоснование

Индивидуально или в группе выбрать идею, сформулировать цель, ожидаемый результат.

Необходимо повторить: критерии SMART, реалистичность.

59. Проектирование архитектуры, выбор библиотек

Схема взаимодействия модулей, выбор технологий (библиотеки, API).

Необходимо повторить: модульность, разделение ответственности.

60–61. Разработка проекта: написание кода, создание репозитория

Реализация функциональности, регулярные коммиты в Git. Промежуточное ревью с учителем.

Необходимо повторить: итеративная разработка, документирование кода.

62. Тестирование, отладка, документирование кода

Написание тестов (хотя бы ручное тестирование), исправление ошибок, добавление docstring и комментариев.

Необходимо повторить: типы ошибок, важность воспроизводимости.

63. Подготовка презентации и демонстрации

Структура презентации: проблема, решение, демонстрация, выводы, перспективы.

Необходимо повторить: навыки публичного выступления..

64. Защита проектов (публичная демонстрация, вопросы)

Каждая группа/участник показывает работающий проект, отвечает на вопросы. Оценивание по рубрике.

Необходимо повторить: аргументация решений, принятие критики.

65. Итоговая рефлексия курса, самооценка

Обсуждение достижений, трудностей, планов на будущее. Заполнение анкеты обратной связи.

Необходимо повторить: перенос знаний в другие предметы и профессиональную сферу.

Резервные часы (66–68)

Резервное время используется для:

- повторения тем, вызвавших затруднения;
- дополнительных консультаций по проектам;
- разбора новых тем по запросу обучающихся (например, веб-фреймворки, асинхронное программирование).

Методическая рекомендация: резерв следует планировать в конце года, но при необходимости использовать ранее для отработки сложных разделов.