

**Федеральное государственное автономное образовательное
учреждение высшего образования
«Московский физико-технический институт
(национальный исследовательский университет)»**

УТВЕРЖДЕНО
Директор

А.В. Малеев

	Рабочая программа дисциплины (модуля)
по дисциплине:	Python продвинутый уровень
по направлению:	Программная инженерия
профиль подготовки:	Разработка программно-информационных систем высшая школа программной инженерии высшая школа программной инженерии
курс:	2
квалификация:	бакалавр

Семестр, формы промежуточной аттестации: 3 (осенний) - Зачет

Аудиторных часов: 60 всего, в том числе:

лекции: 30 час.

семинары: 30 час.

лабораторные занятия: 0 час.

Самостоятельная работа: 12 час.

Всего часов: 72, всего зач. ед.: 2

Программу составил: А.А. Жмуров, канд. физ.-мат. наук, доцент

Программа обсуждена на заседании высшей школы программной инженерии 19.03.2025

Аннотация

Данный курс охватывает углубленные темы программирования на Python, включая работу с данными, асинхронность, многозадачность и веб-разработку. Студенты научатся использовать библиотеки для анализа данных, работать с базами данных, создавать API, а также осvoят методы тестирования и логирования. Курс включает практические задания и примеры из реальных рабочих процессов для применения полученных знаний в профессиональной деятельности.

1. Цели и задачи

Цель дисциплины

Развитие уверенных навыков программирования на Python, позволяющих решать широкий спектр практических задач, разрабатывать эффективный и читаемый код, а также понимать ключевые концепции языка и его экосистемы. Курс сочетает теорию, практику и разбор реальных кейсов, давая актуальные знания для работы с Python.

Задачи дисциплины

- освоить расширенные возможности Python (структуры данных, ООП, исключения, многопоточность);
- научиться писать эффективный и оптимизированный код;
- познакомиться с ключевыми библиотеками (NumPy, Pandas, TensorFlow);
- изучить основы веб-разработки (API, базы данных, ORM);
- освоить тестирование, логирование и паттерны проектирования;
- получить навыки для успешного прохождения собеседований.

2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
УК-2 Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1 Формулирует совокупность взаимосвязанных задач в рамках поставленной цели работы, обеспечивающих ее достижение. Определяет ожидаемые результаты решения поставленных задач
	УК-2.2 Проектирует решение конкретной задачи проекта, выбирая оптимальный способ ее решения, исходя из действующих правовых норм и имеющихся ресурсов и ограничений
ОПК-2 Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности	ОПК-2.1 Способен использовать информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности
ПК-2 Способен формализовать и алгоритмизировать поставленную задачу	ПК-2.1 Способен формализовать и алгоритмизировать поставленную задачу
	ПК-2.2 Владеет методами и приемами формализации и алгоритмизации задач

3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- расширенные возможности Python (ООП, многопоточность, асинхронность, исключения);
- основные библиотеки для анализа данных и машинного обучения (NumPy, Pandas, TensorFlow);
- принципы работы с API, базами данных, ORM;
- подходы к тестированию, логированию и профилированию кода;
- основы паттернов проектирования в Python.

уметь:

- разрабатывать и оптимизировать Python-приложения;
- эффективно работать с большими данными, сериализацией и визуализацией;
- использовать многопоточность, асинхронное программирование и распределенные вычисления;
- писать и документировать тестируемый код;
- разрабатывать и интегрировать API.

владеть:

- методами структурирования и оптимизации кода;
- инструментами автоматизированного тестирования и отладки;
- навыками работы с современными Python-библиотеками и технологиями;
- практиками промышленной разработки на Python.

4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

№	Тема (раздел) дисциплины	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.			
		Лекции	Семинары	Лаборат. работы	Самост. работа
1	Основы Python: типы данных, циклы и функции	2	2		1
2	Структуры данных и ООП	2	2		1
3	Работа с файлами, строками и датами	2	2		1
4	Библиотеки для анализа данных (NumPy, Pandas)	2	2		1
5	Декораторы, замыкания, неймспейсы и типы данных	2	2		1
6	Модули, пакеты, исключения и контекстные менеджеры	2	2		1
7	Многозадачность и асинхронное программирование	3	3		1
8	Тестирование, логирование и паттерны проектирования	3	3		1
9	Работа с базами данных и API	3	3		1
10	Производительность и оптимизация кода	3	3		1
11	Визуализация и работа с большими данными	3	3		1
12	TensorFlow и машинное обучение	3	3		1
Итого часов		30	30		12
Подготовка к экзамену		0 час.			
Общая трудоёмкость		72 час., 2 зач.ед.			

4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 3 (Осенний)

1. Основы Python: типы данных, циклы и функции

Введение в Python, работа с простыми типами данных, циклами, условиями и функциями.

2. Структуры данных и ООП

Изучение списков, кортежей, словарей, классов и принципов ООП: инкапсуляция, наследование, полиморфизм.

3. Работа с файлами, строками и датами

Обработка строк и файлов, манипуляции с датами и временем с использованием библиотеки `datetime`.

4. Библиотеки для анализа данных (NumPy, Pandas)

Использование NumPy и Pandas для обработки и анализа данных.

5. Декораторы, замыкания, неймспейсы и типы данных

Создание декораторов, работа с замыканиями и неймспейсами, аннотации типов и их применение.

6. Модули, пакеты, исключения и контекстные менеджеры

Создание и организация модулей и пакетов, обработка исключений и использование контекстных менеджеров.

7. Многозадачность и асинхронное программирование

Использование многозадачности, потоков, процессов, а также принципов `async/await` и `asyncio`.

8. Тестирование, логирование и паттерны проектирования

Основы тестирования (`unittest`, `pytest`), внедрение логирования и использование простых паттернов проектирования.

9. Работа с базами данных и API

Разработка и интеграция API, использование коннекторов и ORM для работы с базами данных.

10. Производительность и оптимизация кода

Ускорение вычислений, управление памятью и оптимизация Python-приложений.

11. Визуализация и работа с большими данными

Создание визуализаций с помощью Matplotlib и Seaborn, работа с большими данными и потоками (MapReduce).

12. TensorFlow и машинное обучение

Основы работы с TensorFlow для создания и обучения нейронных сетей.

5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Учебная аудитория, оснащенная компьютерами для каждого студента.

6. Перечень рекомендуемой литературы

Основная литература

1. Изучаем Python: программирование игр, визуализация данных, веб-приложения, [руководство] / Э. Мэтиз. — Санкт-Петербург, Питер, 2020. — URL: <https://ibooks.ru/bookshelf/371712/reading> (дата обращения: 24.11.2020). - Полный текст (Режим доступа : из сети МФТИ / Удаленный доступ)

Фонд библиотеки МФТИ:

2. Никитина, Т. П. Программирование. Основы Python для инженеров : учебное пособие для вузов / Т. П. Никитина, Л. В. Королев. — 2-е изд., стер. — Санкт-Петербург : Лань, 2025. — 156 с. — ISBN 978-5-507-50668-2. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/454463>

3. Шелудько, В. М. Язык программирования высокого уровня Python. Функции, структуры данных, дополнительные модули : учебное пособие / В. М. Шелудько ; Южный федеральный университет. - Ростов-наДону ; Таганрог : Издательство Южного федерального университета, 2017. - 107 с. - ISBN 978-5-9275-2648-2. - Текст : электронный. - URL: <https://znanium.com/catalog/product/1021664>

Дополнительная литература

Рекомендуемая литература для самостоятельного изучения

1. Гниденко, И. Г. Технологии и методы программирования: учебное пособие для вузов / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — Москва: Издательство Юрайт, 2022. — 235 с. — (Высшее образование). — ISBN 978-5-534-02816-4. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/489920>

2. Илюшечкин, В. М. Основы использования и проектирования баз данных: учебник для вузов / В. М. Илюшечкин. — Москва: Издательство Юрайт, 2022. — 213 с. — (Высшее образование). — ISBN 978-5-534-03617-6. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/488604>

7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)

-

8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)

Инструменты разработки, библиотеки и фреймворки, системы контроля версий, базы данных, облачные сервисы, тестирование, контейнеризация, асинхронные технологии, анализ данных.

9. Методические указания для обучающихся по освоению дисциплины (модуля)

Успешное освоение курса требует напряжённой самостоятельной работы студента. В программе курса приведено минимально необходимое время для работы студента над темой.

Самостоятельная работа включает в себя:

- проработку учебного материала;
- подготовку к практическим занятиям, выполнение домашних заданий.

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

по направлению:	Программная инженерия
профиль подготовки:	Разработка программно-информационных систем высшая школа программной инженерии высшая школа программной инженерии
курс:	<u>2</u>
квалификация:	бакалавр

Семестр, формы промежуточной аттестации: 3 (осенний) - Зачет

Разработчик: А.А. Жмуров, канд. физ.-мат. наук, доцент

1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
УК-2 Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1 Формулирует совокупность взаимосвязанных задач в рамках поставленной цели работы, обеспечивающих ее достижение. Определяет ожидаемые результаты решения поставленных задач
	УК-2.2 Проектирует решение конкретной задачи проекта, выбирая оптимальный способ ее решения, исходя из действующих правовых норм и имеющихся ресурсов и ограничений
ОПК-2 Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности	ОПК-2.1 Способен использовать информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности
ПК-2 Способен формализовать и алгоритмизировать поставленную задачу	ПК-2.1 Способен формализовать и алгоритмизировать поставленную задачу
	ПК-2.2 Владеет методами и приемами формализации и алгоритмизации задач

2. Показатели оценивания компетенций

В результате изучения дисциплины «Python продвинутый уровень» обучающийся должен:

знать:

- расширенные возможности Python (ООП, многопоточность, асинхронность, исключения);
- основные библиотеки для анализа данных и машинного обучения (NumPy, Pandas, TensorFlow);
- принципы работы с API, базами данных, ORM;
- подходы к тестированию, логированию и профилированию кода;
- основы паттернов проектирования в Python.

уметь:

- разрабатывать и оптимизировать Python-приложения;
- эффективно работать с большими данными, сериализацией и визуализацией;
- использовать многопоточность, асинхронное программирование и распределенные вычисления;
- писать и документировать тестируемый код;
- разрабатывать и интегрировать API.

владеть:

- методами структурирования и оптимизации кода;
- инструментами автоматизированного тестирования и отладки;
- навыками работы с современными Python-библиотеками и технологиями;
- практиками промышленной разработки на Python.

3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

Во время текущего контроля студент должен уметь ответить на следующие вопросы:

1. Работа с файловой системой и модули.
2. Исключения и обработка ошибок.
3. Понятие класса.
4. Регулярные выражения и основы синтаксического разбора.
5. Библиотеки.

6. Библиотека numpy.
7. Вычислительные задачи.
8. Библиотека Pandas.
9. Функции и работа с данными.
10. Продвинутой pandas.
11. Основы парсинга и работы с API.
12. Базовые понятия статистики.
13. Основы визуализации данных в Python (библиотеки matplotlib и seaborn).
14. Случайные события.
15. Случайные величины.
16. Корреляция и корреляционный анализ.
17. Задачи классификации и кластеризации.
18. Доверительные интервалы.
19. Статистическая проверка гипотез для несвязанных выборок.
20. Статистическая проверка гипотез для связанных выборок.
21. А/В тесты и как их проводить.
22. Кейс-стади.
23. Работа с локальным репозиторием в Git.
24. Работа с удаленным репозиторием через GitHub.
25. Командная работа в Git и GitHub.
26. Объекты и классы.
27. Взаимодействие между ними.
28. Наследование, инкапсуляция и полиморфизм.
29. Открытие и чтение файла.
30. Запись в файл.
31. Работа с разными форматами данных.
32. Работа с библиотекой requests, http-запросы.
33. Работа с классами на примере API VK.
34. Select-запросы, выборки из одной таблицы.
35. Продвинутой выборка данных.
36. Работа с PostgreSQL из Python.
37. Python и базы данных. ORM.
38. Модули, пакеты, импорты в Python.
39. Регулярные выражения.
40. Веб-скрапинг.
41. Итераторы, генераторы.
42. Обработка запросов и шаблоны.
43. Работа с ORM.
44. API на примере Django REST framework.
45. Разделение доступа в DRF.
46. Тестирование Django-приложений с использованием Pytest.
47. Установка пакетов в Ubuntu. Маршрутизация.
48. Связывание хостинга файлов и запуска веб-приложения.
49. Использование дополнительных файлов для работы веб-приложения.

Во время занятий могут проходить интерактивные обсуждения в чатах курса, что будет являться домашним заданием. Успешное выполнение всех заданий по курсу и выполнение контрольных срезов знаний дает преимущество на экзамене.

4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

Перечень контрольных вопросов для сдачи зачета:

1. Какие типы данных поддерживает Python и как с ними работать?

2. В чём разница между списками и кортежами в Python?
3. Как работает принцип замыканий в Python?
4. Что такое декораторы и как их использовать?
5. Как создаются и используются классы в Python?
6. Что такое и как использовать модули и пакеты?
7. Как работать с исключениями и контекстными менеджерами?
8. Что такое итераторы и генераторы?
9. Как работает многозадачность в Python (поток, процессы)?
10. Какие библиотеки Python используются для работы с данными и как они применяются?
11. В чём принцип работы асинхронности в Python?
12. Как тестировать Python-код?
13. Как работать с базами данных в Python?
14. Какие методы сериализации данных существуют в Python?
15. Как создаются и используют API в Python?

Критерии оценивания

«зачтено» - выставляется обучающемуся, если он показал всесторонние, систематизированные знания учебной программы дисциплины и умение применять их на практике при решении конкретных задач (даже если при этом были допущены небольшие неточности), обосновывает принятые решения;

«не зачтено» - не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных понятий дисциплины и не умеет использовать полученные знания при решении типовых практических задач.

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Зачет проводится по итогам текущей успеваемости и сдачи практических заданий, предусмотренных программой дисциплины, путем организации специального опроса, проводимого в устной или письменной форме, а также защитой выпускного проекта.